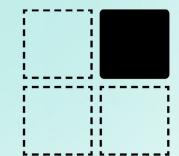




# Our Adventure building a massively scalable C++ remote compiler cloud

## Damien Buhl



**CODE RECKONS**

Science to the CORE

# Our Adventure building a massively scalable C++ remote compiler cloud

CPPP 2021 – 3<sup>rd</sup> December 2021

[www.tipi.build](http://www.tipi.build)

Damien Buhl  
[damien@tipi.build](mailto:damien@tipi.build)  
+41 (0) 78 984 08 13



Tipi

# Our Founders



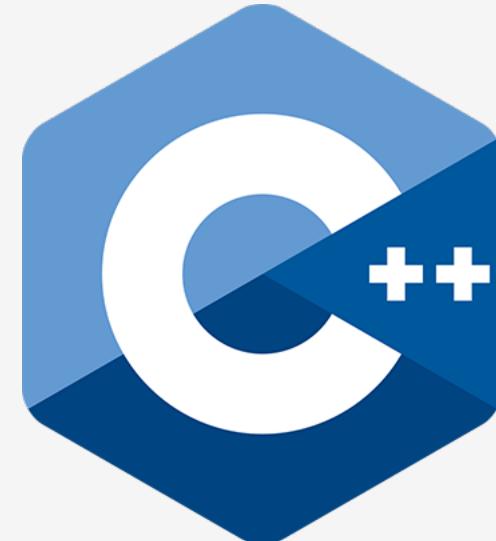
**Damien Buhl**

CEO & CTO



**Yannic Staudt**

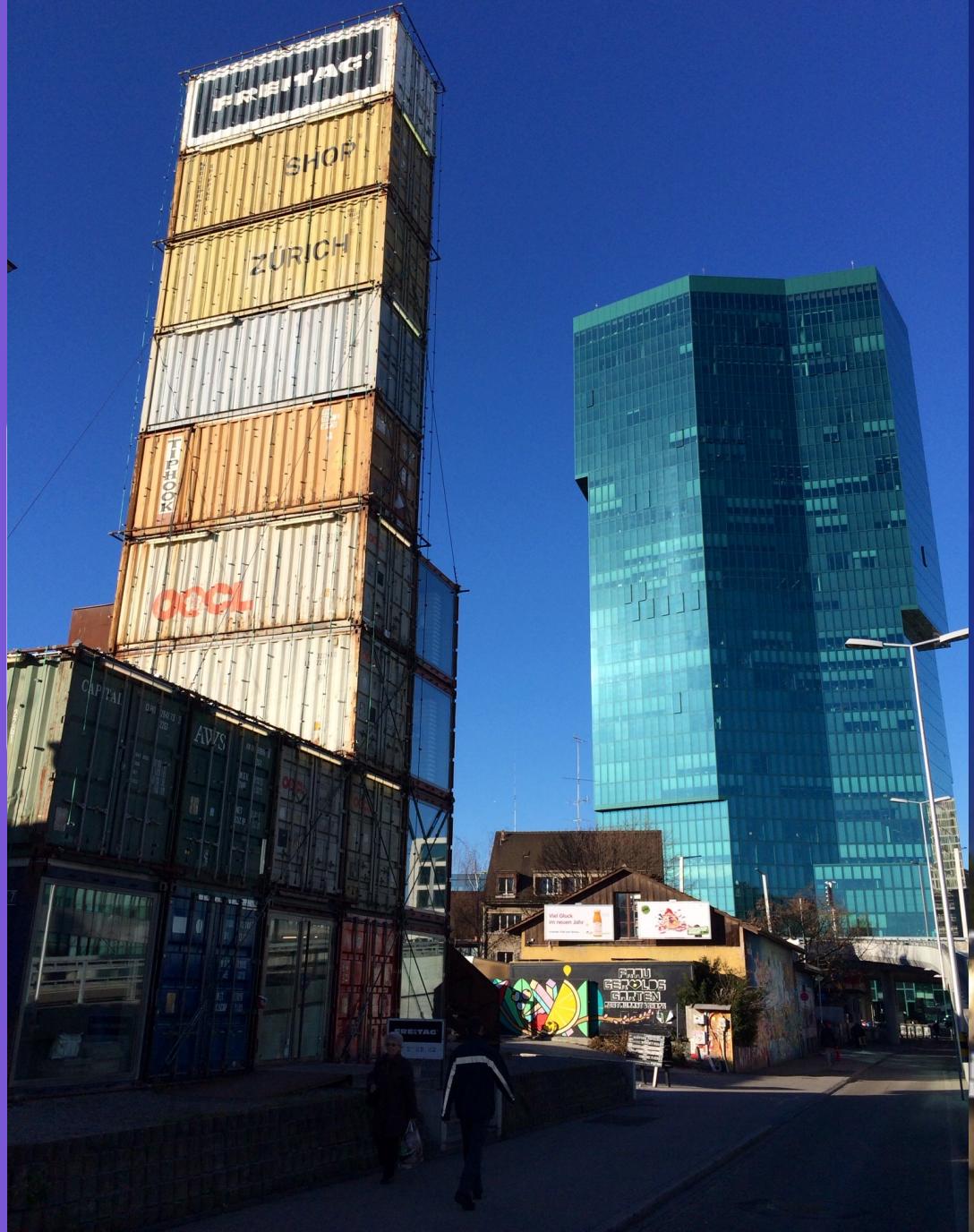
COO & Business Development



# Our Adventures building a massively scalable C++ remote compiler cloud

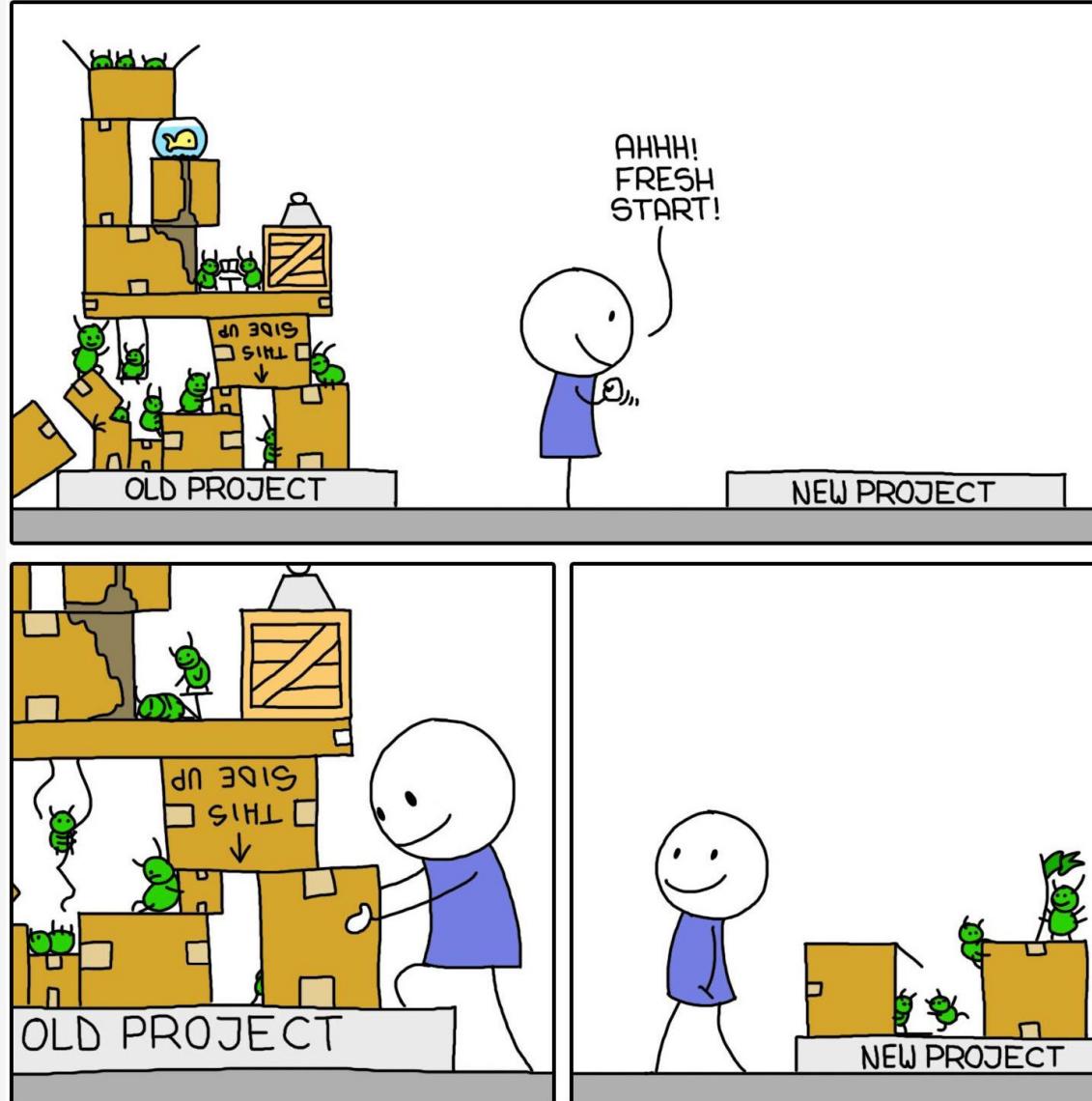
## Agenda

- Why C++ for Webapps ?
- tipi.build software architecture
  - JWT
  - REST
  - OpenAPI
  - ORM

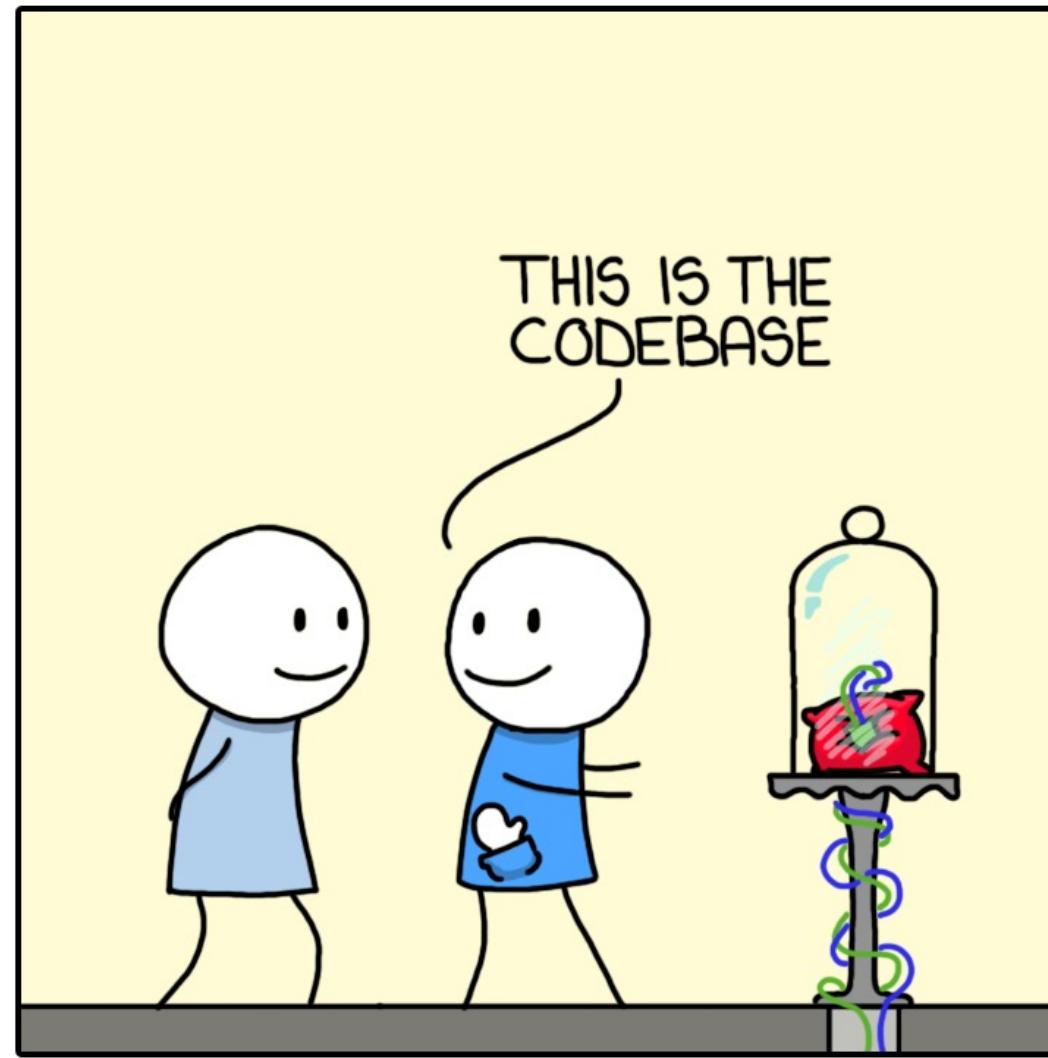


# 3 problems with C & C++

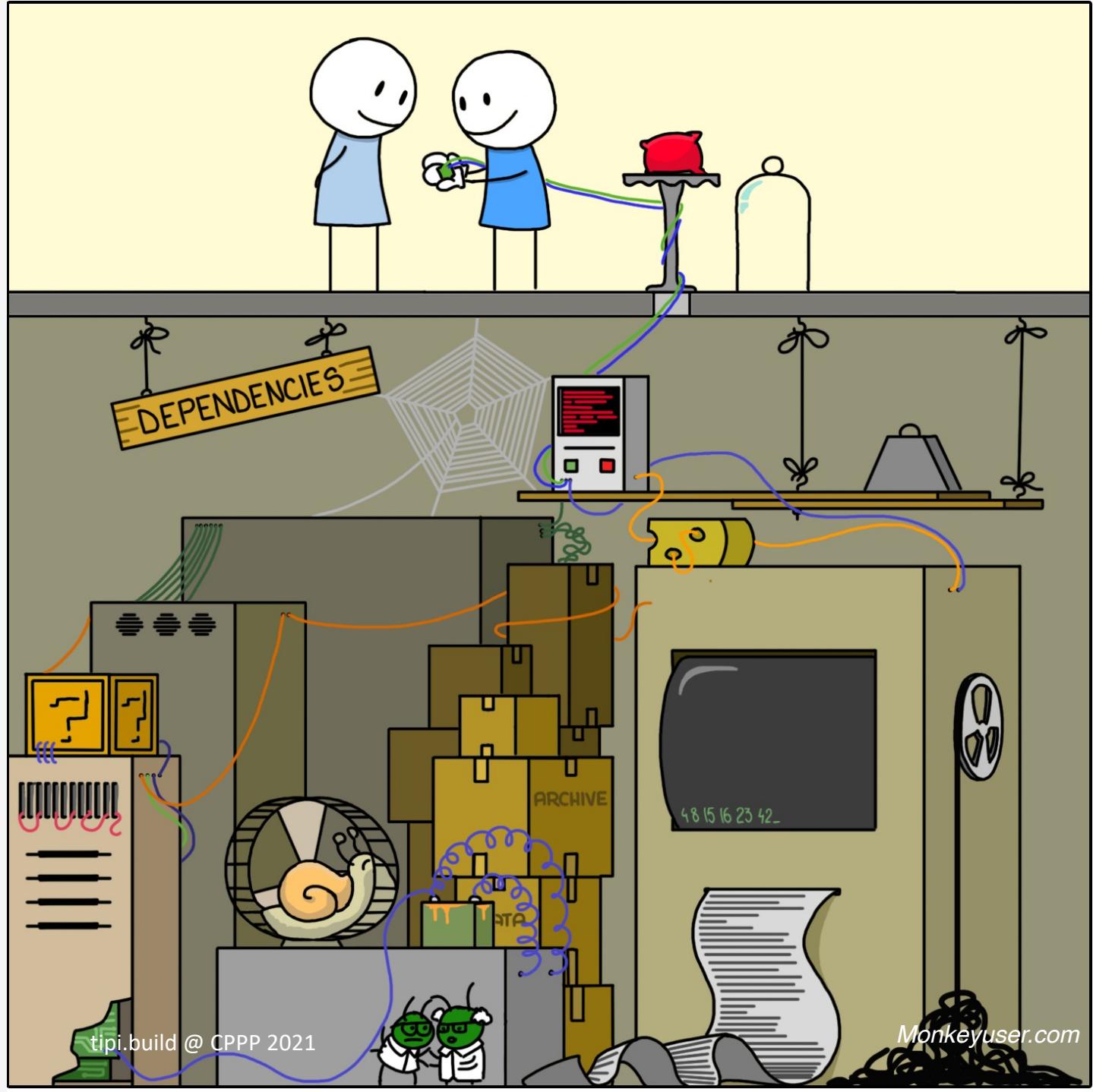
# Code Reuse



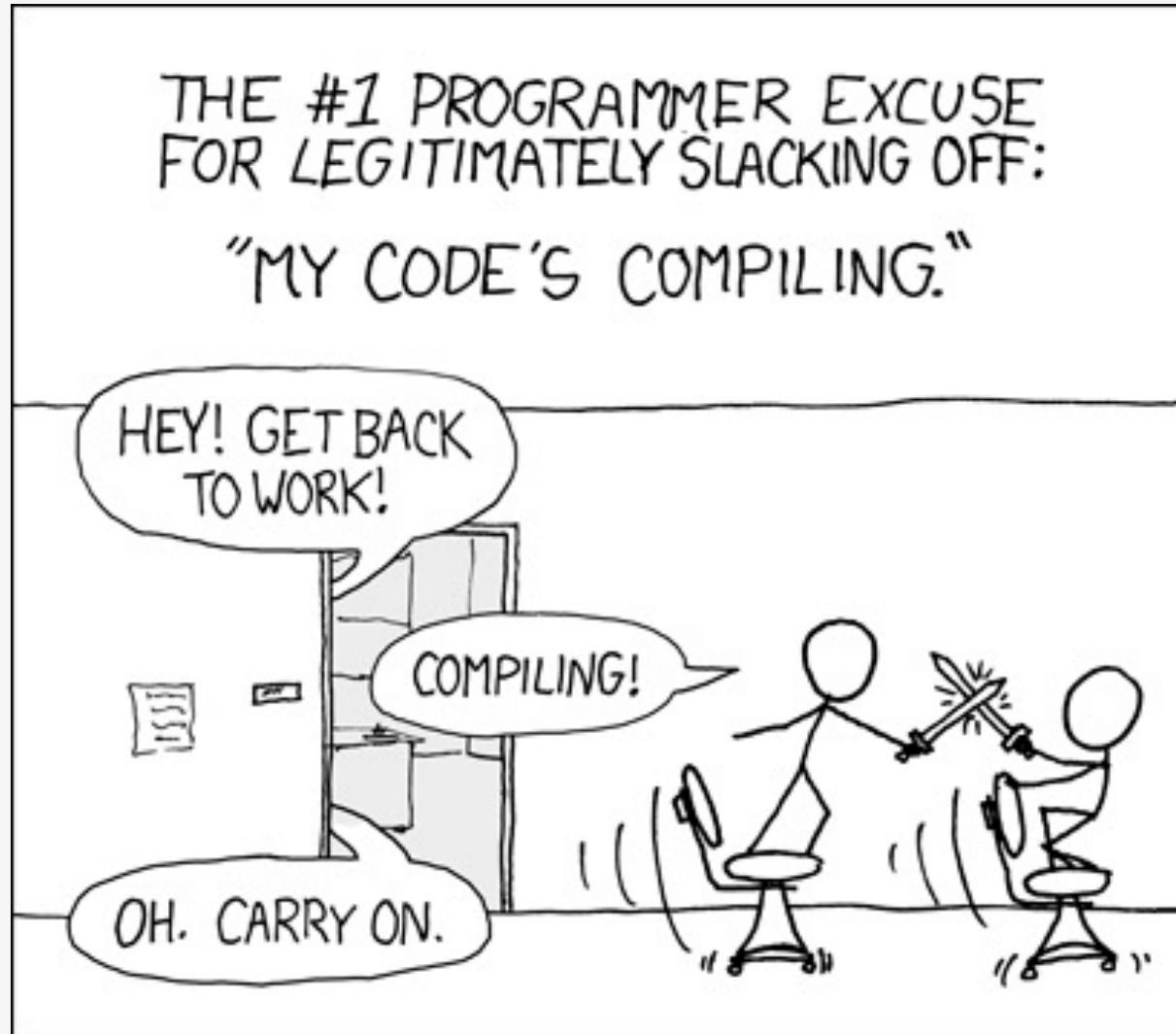
# Dependencies & Environments



*Comics from Monkeyuser.com*



# Build & Deploy Time



Comics from [xkcd.com](http://xkcd.com)

# Join the open beta



**Tipi.build**

Build

Dependency Management  
+  
Superfast cloud powered C & C++ builds.



**Tipi.run**

Run & CI/CD

Machines sized right for C & C++

Join the Beta !

*The fastest native edit-build-test cycle*

Sign-in now :

**<https://tipi.build/>**

**Enable Promo Code :**

**tipi::build(CPPP)**

3 Skip Ad ►|

*"It could definitely change the whole ecosystem!"*

Arthur Sonzogni – C++ Software Developer

# Join the open beta



**Tipi.build**

Build

Dependency Management  
+  
Superfast cloud powered C & C++ builds.



**Tipi.run**

Run & CI/CD

Machines sized right for C & C++

Join the Beta !

*The fastest native edit-build-test cycle*

Sign-in now :

**<https://tipi.build/>**

**Enable Promo Code :**

**tipi::build(CPPP)**

2 Skip Ad ►|

*"It could definitely change the whole ecosystem!"*

Arthur Sonzogni – C++ Software Developer

# Join the open beta



**Tipi.build**

Build

Dependency Management  
+  
Superfast cloud powered C & C++ builds.



**Tipi.run**

Run & CI/CD

Machines sized right for C & C++

Join the Beta !

*The fastest native edit-build-test cycle*

Sign-in now :

**<https://tipi.build/>**

**Enable Promo Code :**

**tipi::build(CPPP)**

1 Skip Ad ►|

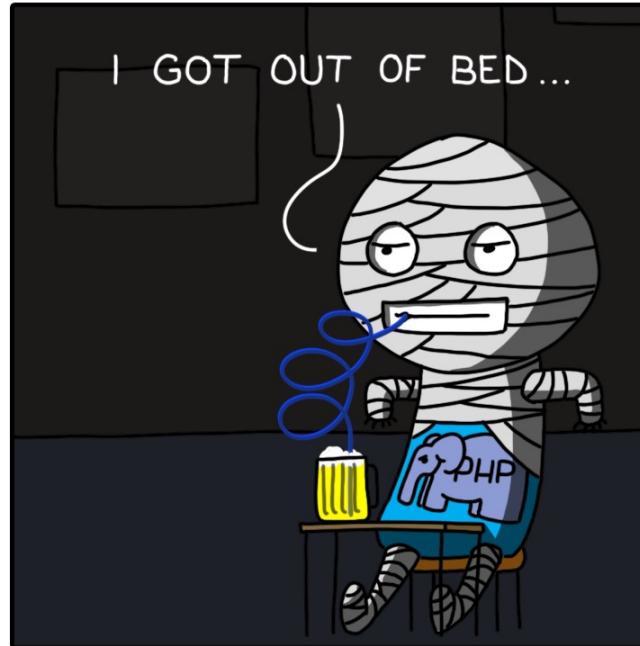
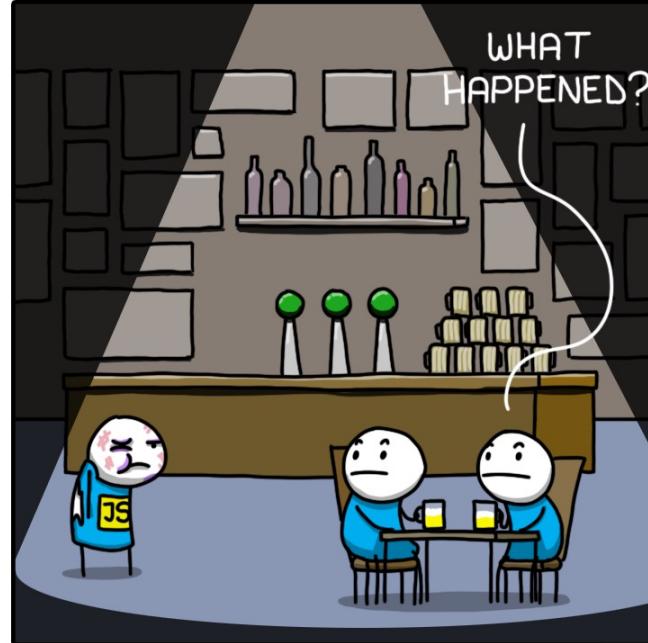
*"It could definitely change the whole ecosystem!"*

Arthur Sonzogni – C++ Software Developer

# Why C++ for webapps ?

## MASOCHISM

MONKEYUSER.COM



# Seriously ! Why C++ for webapps ?

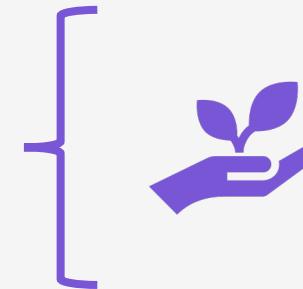
# Carbon footprint

Enabling environmentally friendly software



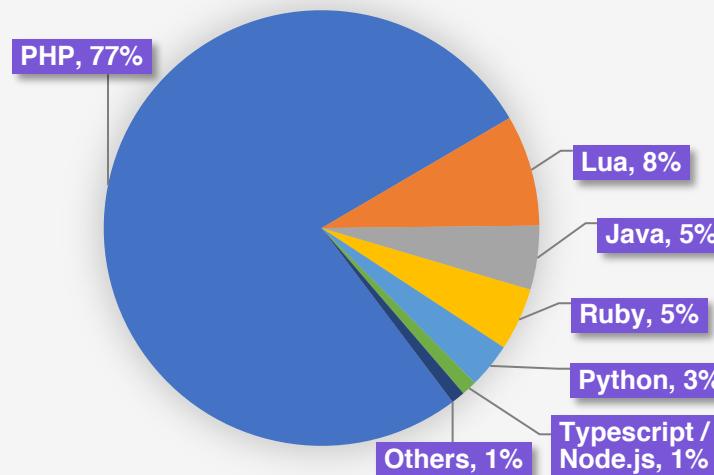
1.32bn CO<sub>2</sub> tons

Yearly Global Emissions by Internet <sup>1</sup>



47'000 CO<sub>2</sub> tons

Yearly Emissions addressable <sup>2</sup> with C++



186M

Of all internet requests by seconds served by non-native technologies globally <sup>3</sup>

<sup>1</sup> The Shift Project, Lean ICT, March 2019

<sup>2</sup> Cisco Annual Internet Report 2020 + HTTP Archive, State of The Web 11.2020

<sup>3</sup> W3Techs, server-side programming languages distribution, top 10 Millions websites, 11.2020

# Carbon footprint

Case study on a typical web application

To serve 43 000 web requests / seconds with common  
cloud hardware :



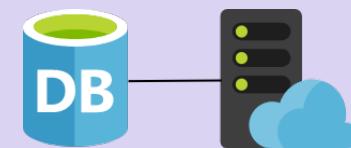
Dell R440 Xeon Gold  
1480 kWh Energy Demand / year

State of the art (e.g. PHP, Python, TS, Ruby)



12'880 kgCO<sub>2</sub>e / year

C++



1'840 kgCO<sub>2</sub>e / year

Dell estimated product carbon footprint & single query TechEmpower Web Framework Benchmark (May 2020)

# Why C++ for webapps ?

*Efficiency !*

# First part of our journey : Microservices vs Monoliths

# Microservices vs Monoliths

## Monoliths

- Vertical scalability only
- Availability issues
- + Simplicity
- + Everything in the same memory space

# Microservices vs Monoliths

## Microservices

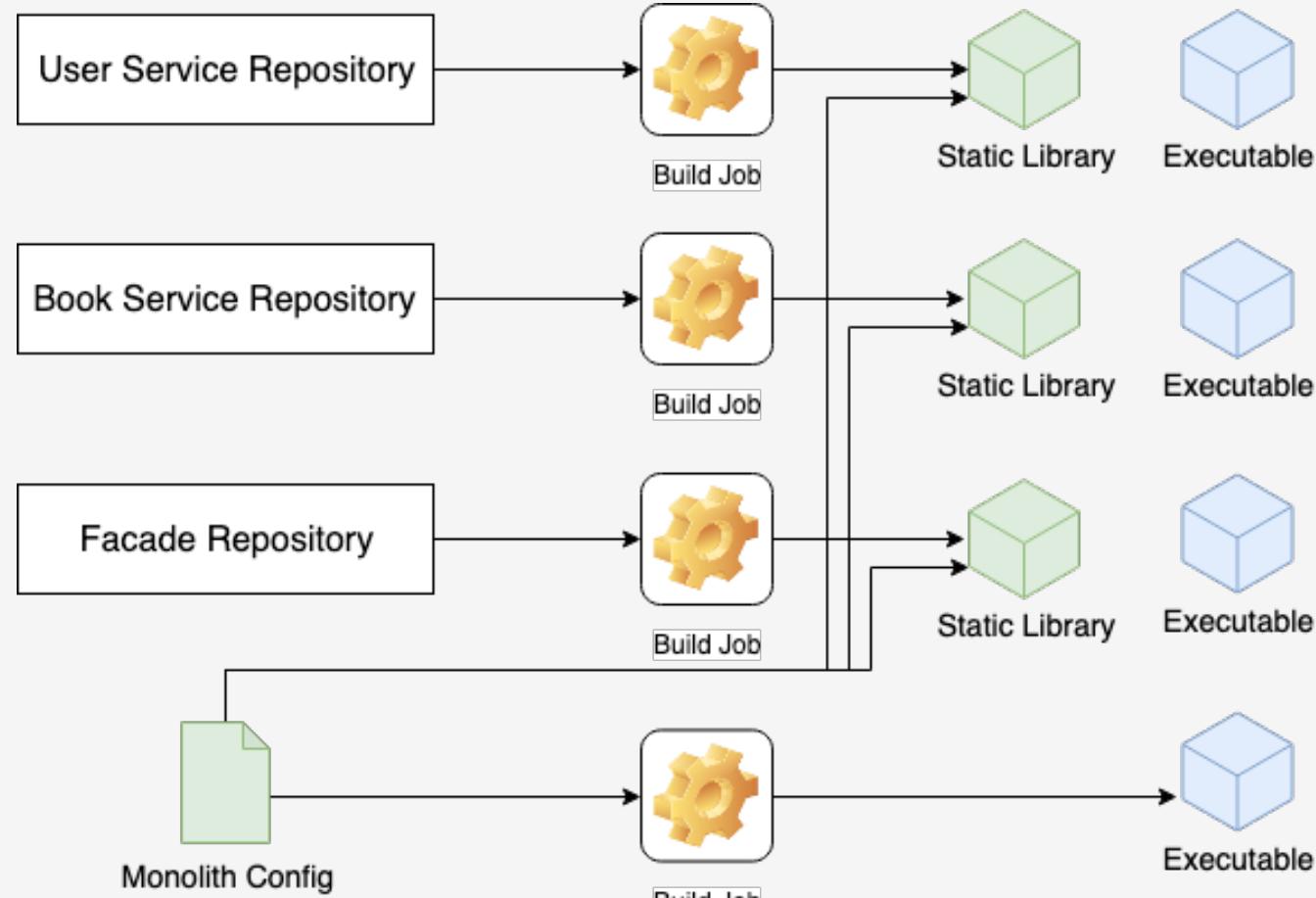
- Lot of Marshalling code between services
  - Shared state data races
- ~ Prone to Conway's Law
- + Horizontal scalability
  - + High Availability
  - + Degraded operation mode

# Monolithization

Term coined by Leonid Stryzhevskyi (@lganzzzo)

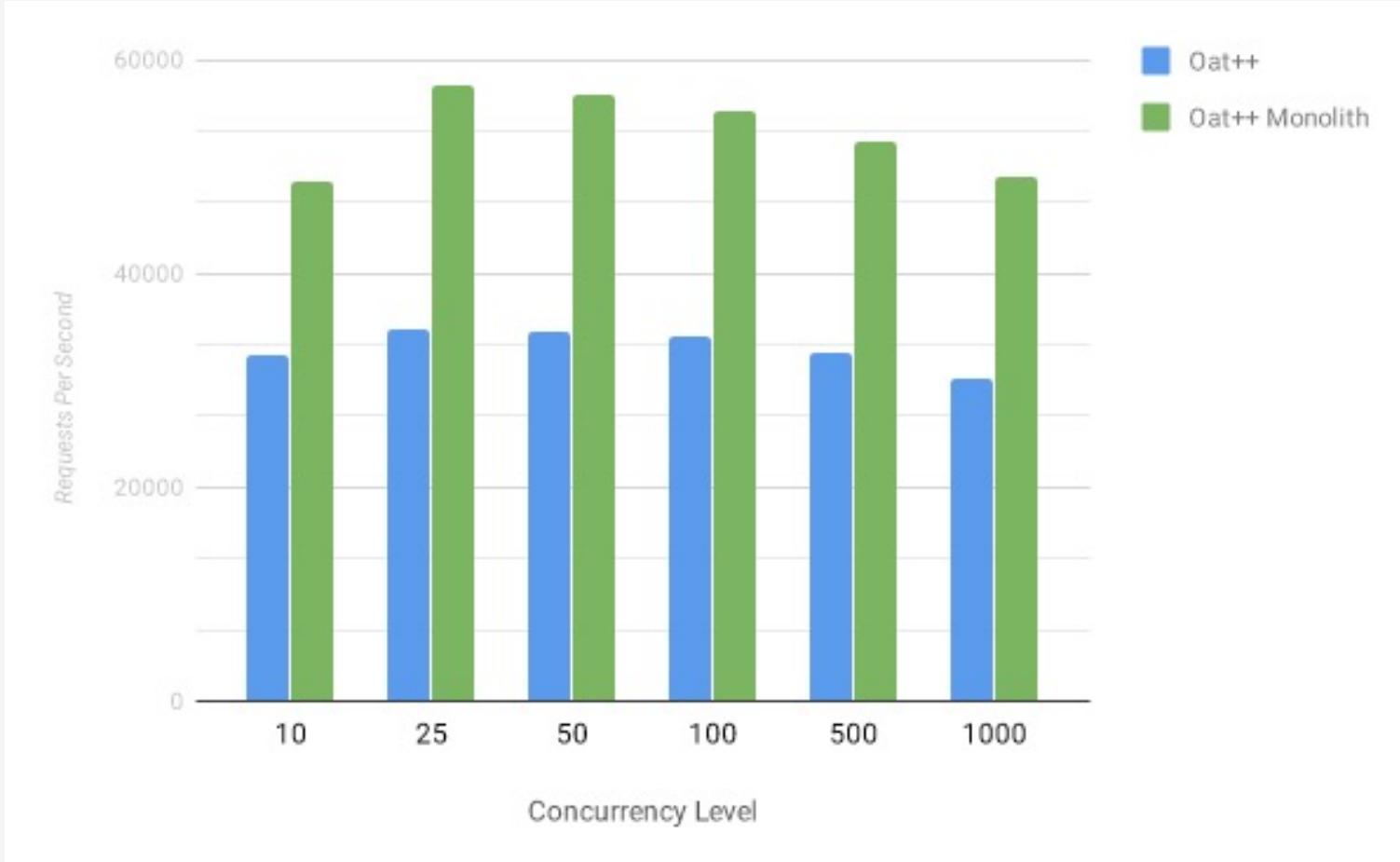
<https://oatpp.io/docs/monolithization/>

# Monolithization

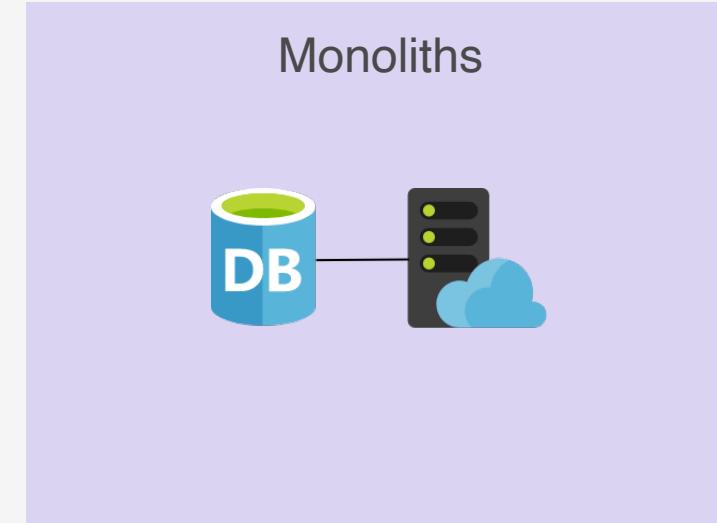
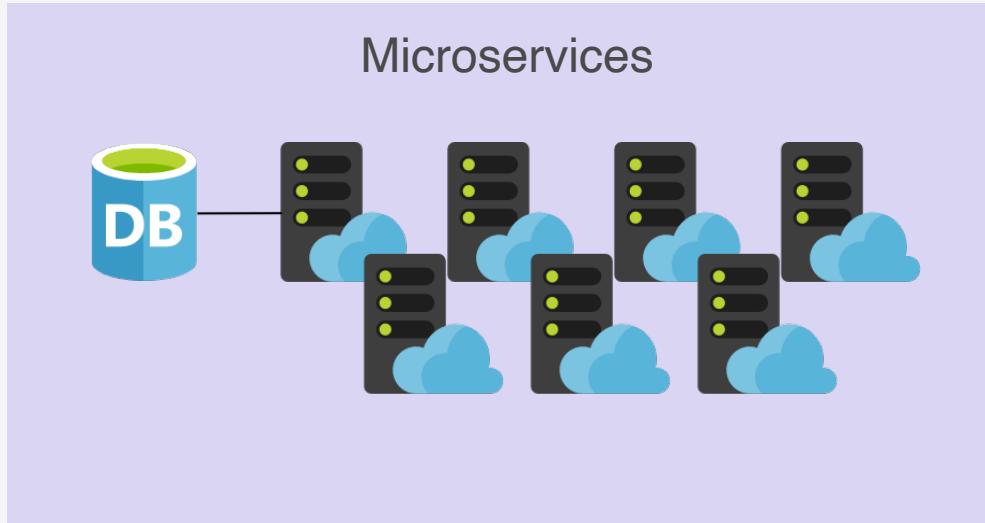


Source : oatpp.io

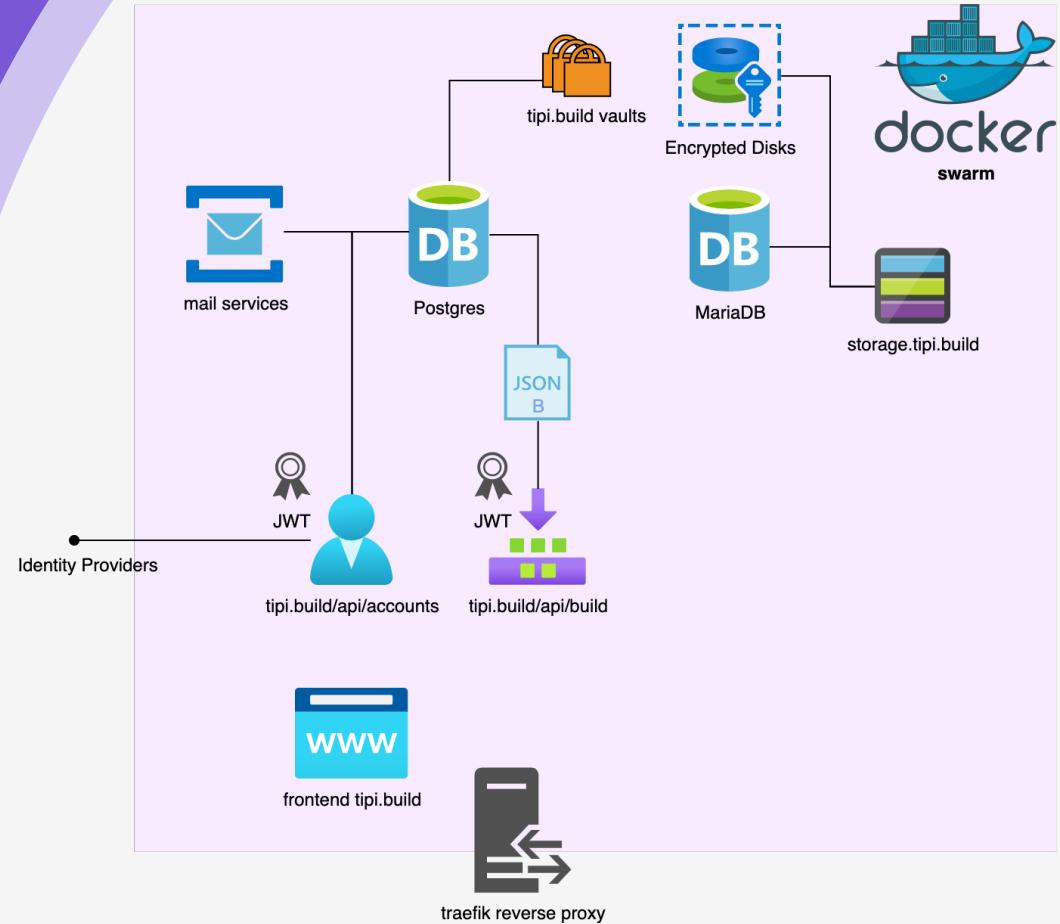
# Monolithization

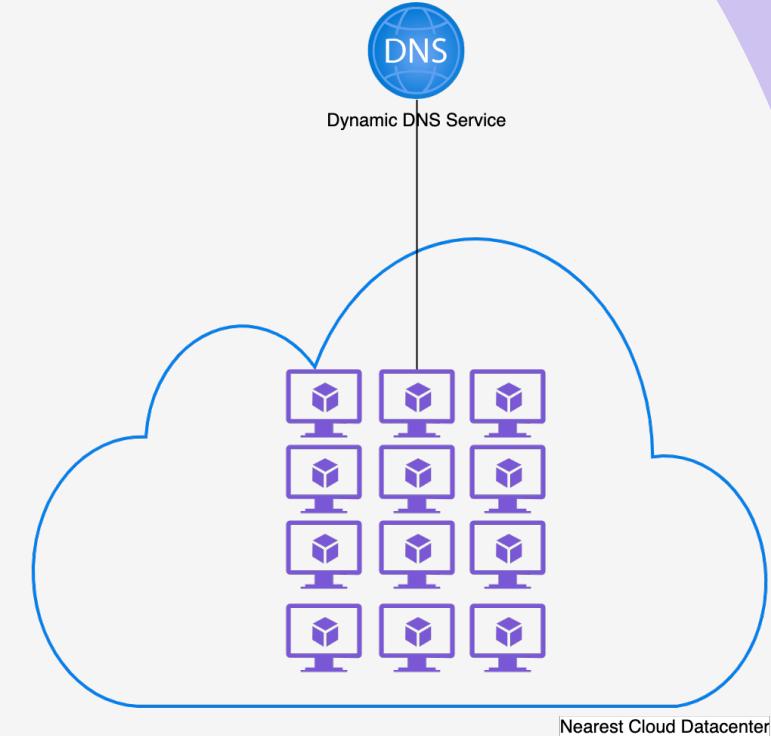
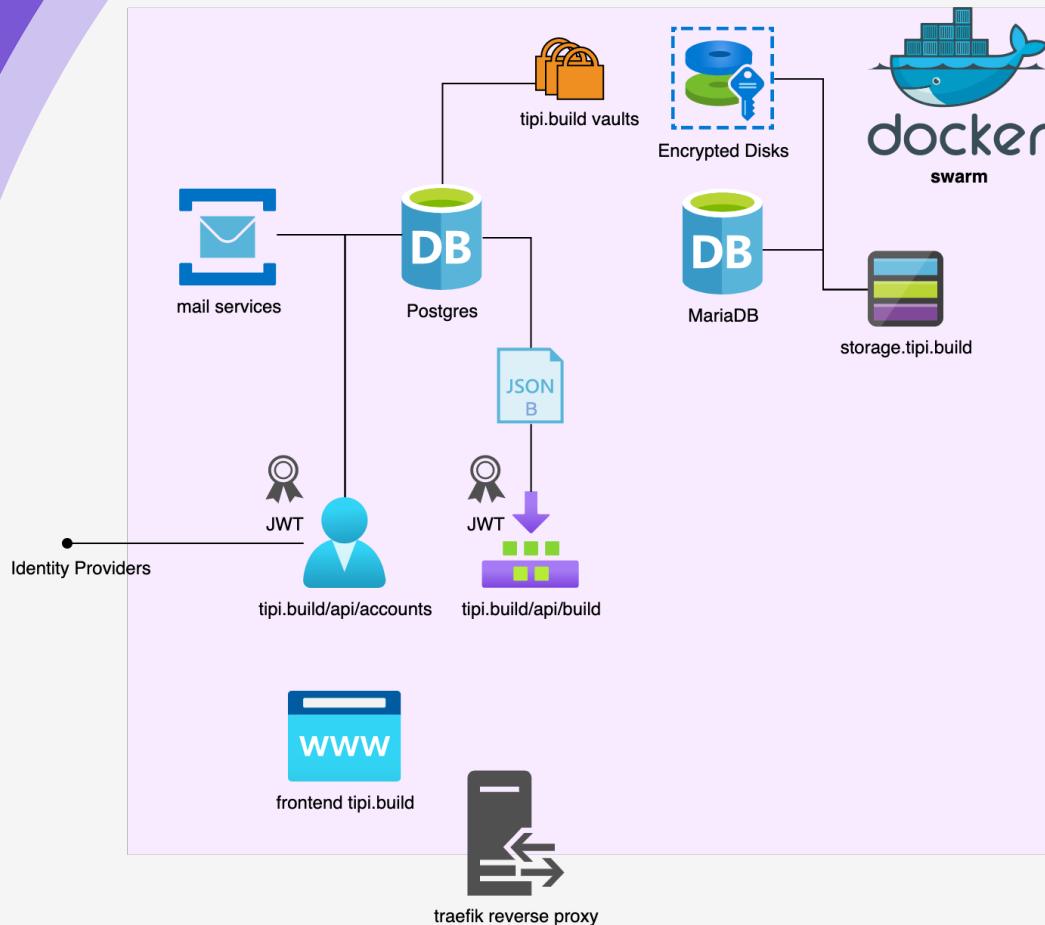


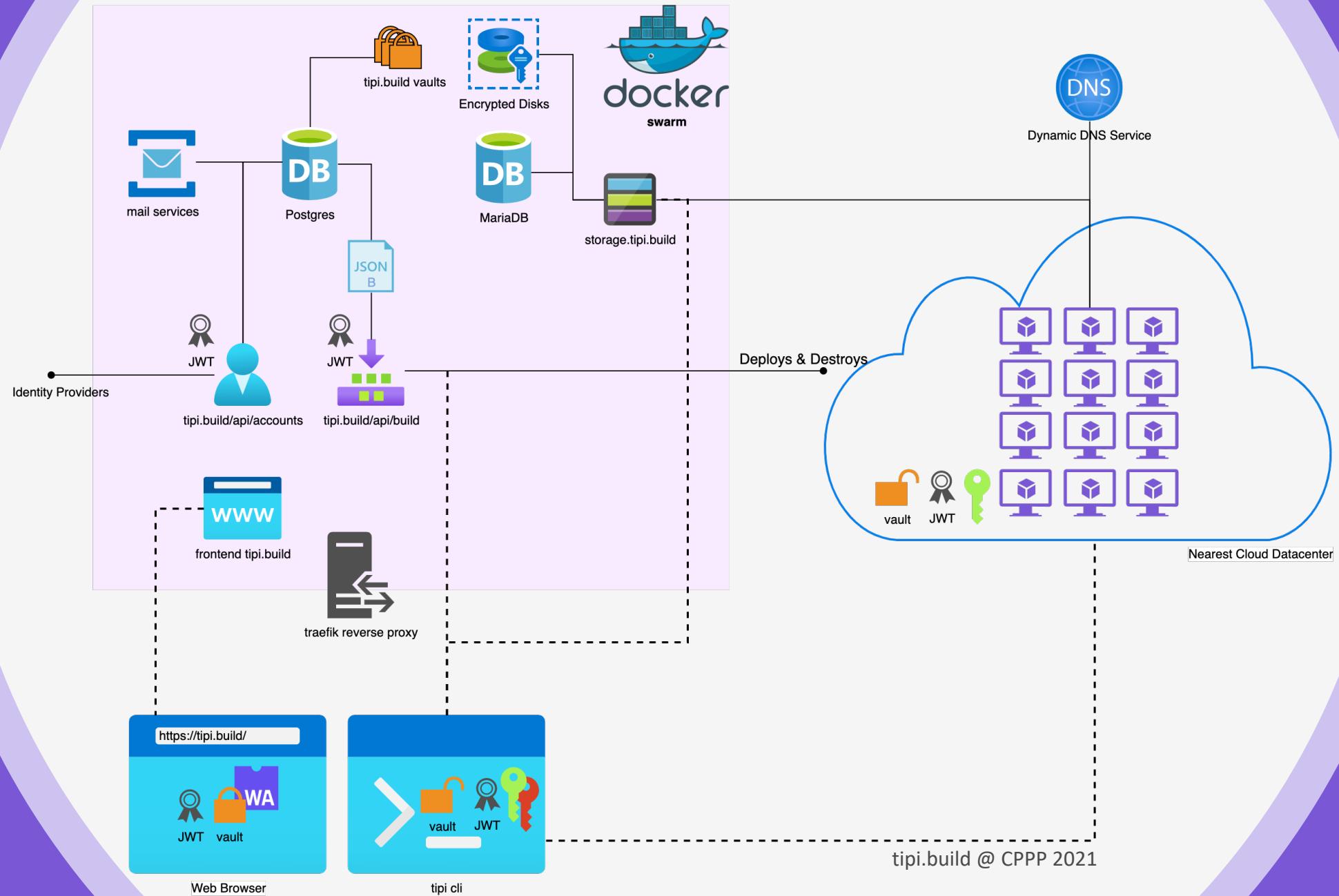
Source : oatpp.io



# How is Tipi.build built ?







# How is **Tipi.build** built ?

C++

# Technology Stack

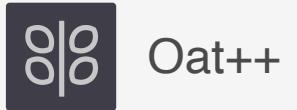




**Tipi.build** helps reusing code.  
Reuse code !

# C++

## Technology stack



```
1 {  
2   "oatpp/oatpp" : { "@" : "1.3.0" }  
3 }
```

Oatpp-starter at <https://github.com/tipi-build/oatpp-starter>

C++

# Technology stack



Oat++

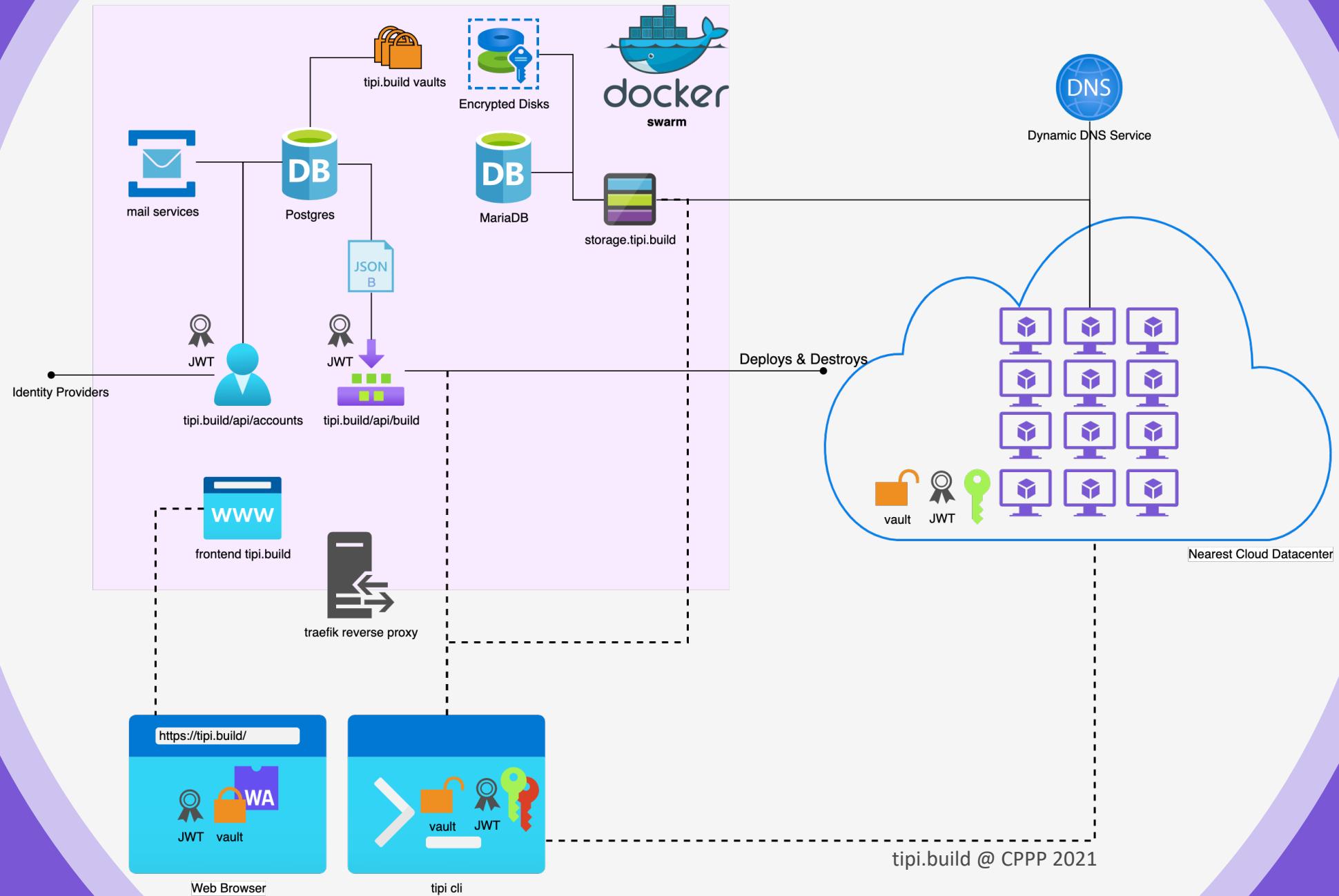


WebAssembly



NuxtJS







# We dogfood but...



## Tipi.build needs you !

<https://tipi.build/>

**Enable Promo Code :**  
**tipi::build(CPPP)**

*Be part of the C++ build revolution !*

*“It could definitely change the whole ecosystem!”*  
Arthur Sonzogni – C++ Software Developer



Tipi.build

is

built with



Tipi.build

# REST APIs

# REST : Intro

## REST

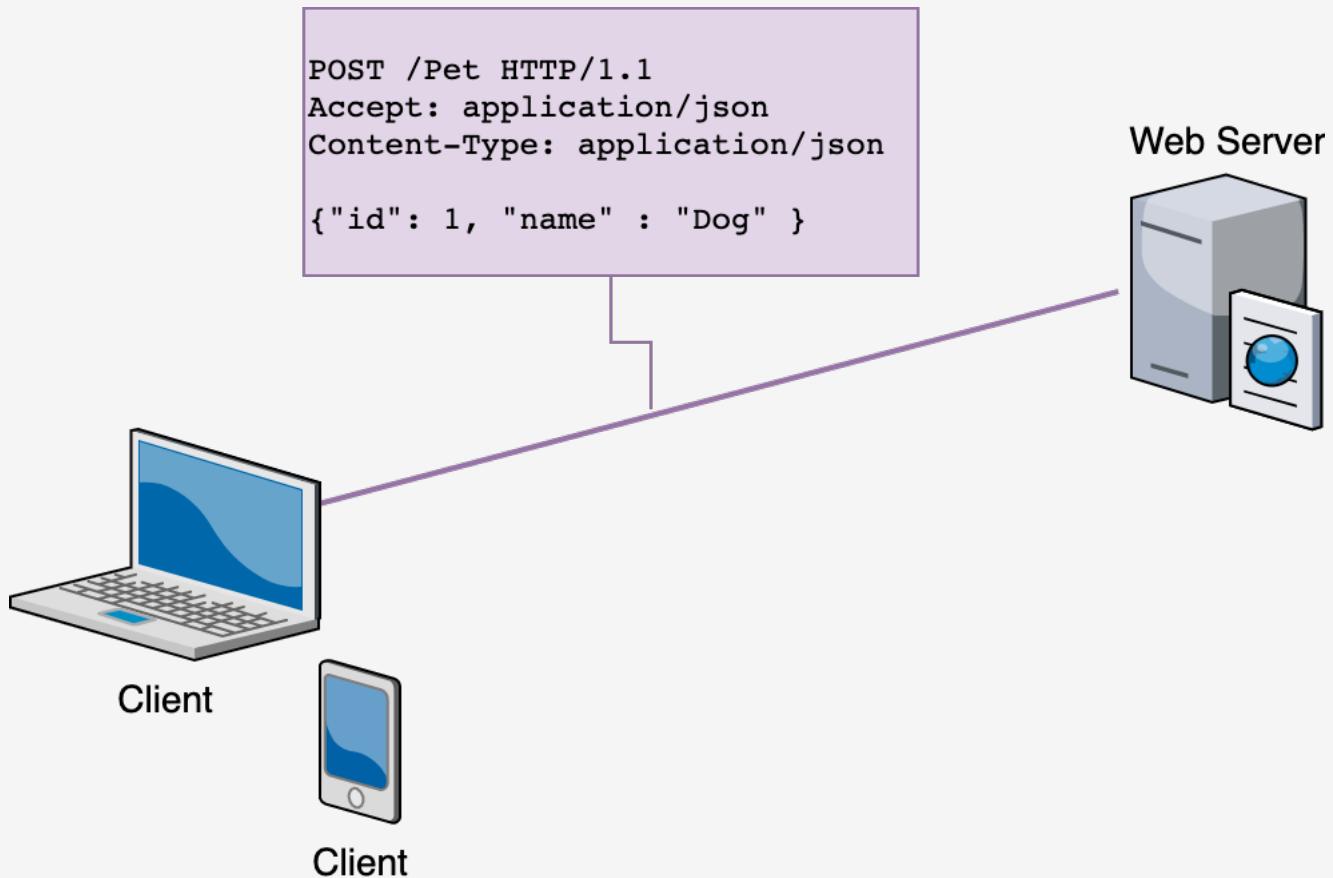
Representational state transfer

- Accessing resources using URIs
- Resources encapsulate entities
- Resources are manipulated
  - CRUD
  - through hypertext representations
  - transferred in messages
- Strong decoupling of client and server  
“anarchic scalability” & “low-entry barrier”

# RESTful webservices

- A base URI
- HTTP methods
  - e.g. GET, POST, PUT, DELETE
- media type that defines state transition data elements
  - e.g. application/json; charset=utf-8
- URI segments to identify resources

# RESTful webservices



# REST Server

## Oat++ RESTful Webservice

```
1 class Controller
2   : public oatpp::web::server::api::ApiController {
3
4   public:
5     ENDPOINT("GET", "/", root) {
6       auto dto = Dto::createShared();
7       dto->statusCode = 200;
8       dto->message = "Hello from tipi.build!";
9       return createDtoResponse(Status::CODE_200, dto);
10    }
11
12 };
```



```
1 class Dto : public oatpp::DTO {
2   DTO_INIT(Dto, DTO)
3
4   DTO_FIELD(Int32, statusCode);
5   DTO_FIELD(String, message);
6 };
```

Data Transfer Object

# OpenAPI REST

# Oat++ OpenAPI Server

## OpenAPI REST

```
 1 class MyController : public oatpp::web::server::api::ApiController {  
 2     public:  
 3         ENDPOINT_INFO(root) {  
 4             info->summary = "Greets the world! ";  
 5  
 6             info->addResponse<Object<Dto>>(Status::CODE_200, "application/json");  
 7             info->addSecurityRequirement("bearer");  
 8         }  
 9         ENDPOINT("GET", "/", root) {  
10             auto dto = Dto::createShared();  
11             dto->statusCode = 200;  
12             dto->message = "Hello World!";  
13             return createDtoResponse(Status::CODE_200, dto);  
14         }  
15     };
```

To learn more about OpenAPI Clients  
with C++20

Watch our CppCon2021 talk

[Our Adventures With REST API in C++ : Making it Easy](#)



**JWT**

# JSON Web Token

## JWT: HTTP Bearer Auth



```
1 namespace tipi::accounts {
2     struct jwt {
3         size_t iat;
4         size_t exp;
5         std::string aud;
6         std::string iss;
7         //! subject is user id
8         std::string sub;
9     };
10 }
11
12 namespace tipi::accounts {
13     struct jwt_access : public jwt {
14         std::string nick;
15         std::string email;
16         std::vector<std::string> organizations;
17         std::string grant;
18     };
19 }
```



JWT

# JSON Web Token

## Adding support to Oat++

```
1 {  
2   "arun11299/cpp-jwt": {  
3     "@": "master",  
4     "requires": {  
5       "nlohmann/json": { "@": "v3.1.2" },  
6       "platform": [  
7         "OpenSSL::+SSL",  
8         "OpenSSL::+Crypto"  
9       ]  
10    }  
11  }  
12 }
```

# Oat++ BearerAuthorizationHandler

```
1 class TipiAuthorizationHandler
2   : public oatpp::web::server::handler::BearerAuthorizationHandler {
3
4   public:
5     std::shared_ptr<AuthorizationObject> authorize(const oatpp::String &token) override {
6       using namespace jwt::params;
7
8       std::error_code ec;
9       jwt::decode(token, algorithms({"ES384"}), ec, verify(true), secret(pub_key));
10
11      return (!ec) ?
12          std::make_shared<AuthorizationObject>()
13          : nullptr;
14    }
15  };
16
17 //... In Controller Constructor
18
19 setDefaultAuthorizationHandler(std::make_shared<TipiAuthorizationHandler>());
```

# ORM

## *Object-Relational Mapping*

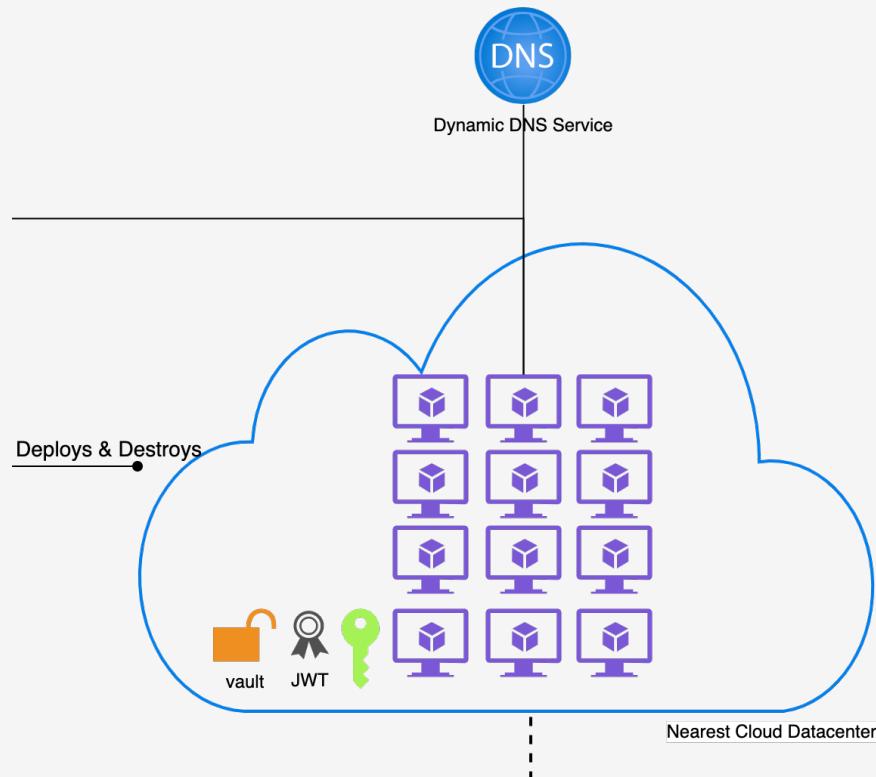
# ORM

*Thing that maps SQL Tables to  
Classes*

# NoORM for NoSQL

We'll store objects directly

# tipi.build machines



# JSON and a bit of SQL

```
1 struct machine_t {
2     std::string id{ to_string(boost::uuids::random_generator()()) };
3
4     cloud_vendor_t cloud_vendor{};
5     machine_type_t machine_type{};
6
7     /// the running environment
8     std::string environment_name;
9     std::string environment_hash;
10
11    /// when the machine was kept alive by an active user session at last
12    std::chrono::milliseconds::rep last_keep_alive{};
13
14    std::string job_id{};
15    execution_status_t status{execution_status_t::STOPPED};
16
17    std::optional<std::string> fqdn{};
18
19    /// the machine name in AZ dns (without domain)
20    std::optional<std::string> dns_name{};
21    std::string address{};
22    std::size_t port{};
23
24    std::optional<std::string> subscription_id{};
25 };
```

```
1 CREATE TABLE IF NOT EXISTS machine(
2     data jsonb);
3
4 CREATE INDEX machine_idx
5     ON machine
6     USING gin(data jsonb_path_ops);
7
8 CREATE UNIQUE INDEX machine_uuid_idx
9     ON machine(((data->>'id')));
```

# JSON and a bit of SQL

```
1 QUERY(create_machine,
2     R"(INSERT INTO machine (data) VALUES (CAST(:data_to_insert as json)) RETURNING *;)",
3     PREPARE(true),
4     PARAM(oatpp::String,data_to_insert))
5
6 tipi::build::service::machine_t create_machine(const machine_t &machine) {
7     tipi::build::service::machine_t machine_for_db = machine;
8
9     std::string machine_string = pre::json::to_json(machine_for_db).dump();
10    auto dbResult = m_database->create_machine(machine_string.c_str());
11
12    OATPP_ASSERT_HTTP(dbResult->isSuccess(), Status::CODE_500, dbResult->getErrorMessage());
13    return machine_for_db;
14 }
```

# To serialize JSON like the cool kids with C++20

## Watch our CppCon2021 talk

[Our Adventures With REST API in C++ : Making it Easy](#)

<https://github.com/tipi-build/openapipp/>

# C++20 better [de]serialization

```
1 #include <pre/json/from_json.hpp>
2
3 struct person {
4     pre_json_key(std::string, name);
5     pre_json_key(std::size_t, age);
6     pre_json_key(std::string, address);
7 };
8
9 //...
10 auto person = pre::json::from_json<person>(R"( 
11     { "name" : "Edouard",
12       "age" : 31,
13       "address" : "hello@tipi.build"
14     }
15 )");
```

# JSONB advantages

- Postgres JSONB performance
  - Faster than mongodb
  - Same speed as columns
- Simplicity
- Who needs data migration when you have std::optional ?



**Tipi.build** deploys many  
manycores machines !

# tipi.build environments



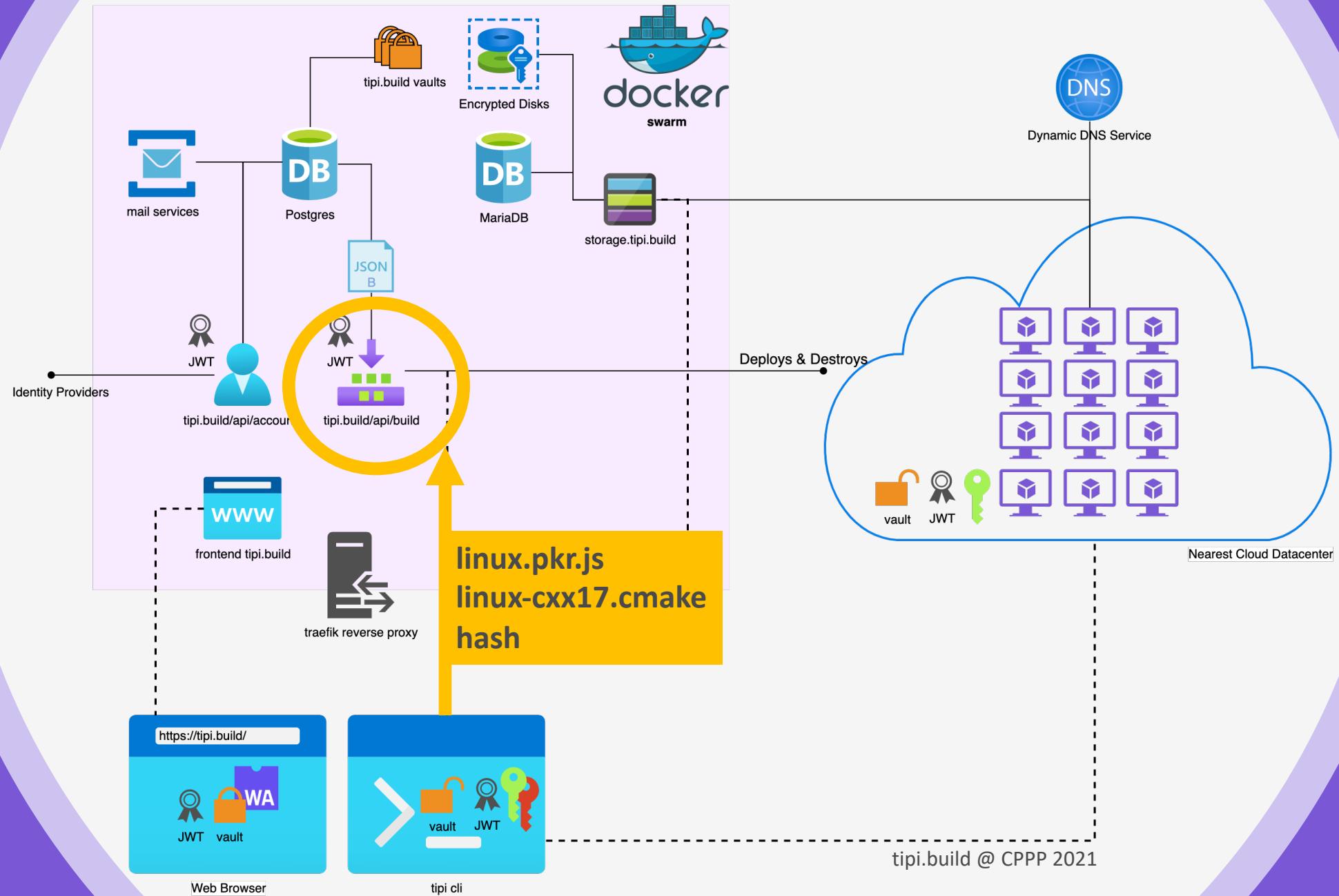
linux-cxx17.cmake

```
1 if(DEFINED POLLY_CLANG_CXX17_CMAKE_)
2   return()
3 else()
4   set(POLLY_CLANG_CXX17_CMAKE_ 1)
5 endif()
6
7 include("${CMAKE_CURRENT_LIST_DIR}/utilities/polly_init.cmake"
8
9 polly_init(
10   "clang / c++17 support"
11   "Unix Makefiles"
12 )
13 include("${CMAKE_CURRENT_LIST_DIR}/utilities/polly_common.cma
14
15 include("${CMAKE_CURRENT_LIST_DIR}/compiler/clang.cmake")
16 include("${CMAKE_CURRENT_LIST_DIR}/flags/cxx17.cmake")
```



linux.pkr.js

```
1 {
2   "variables": { },
3   "builders": [
4     {
5       "type": "docker",
6       "image": "tipibuild/tipi-ubuntu",
7       "commit": true
8     }
9   ],
10  "post-processors": [
11    {
12      "type": "docker-tag",
13      "repository": "linux",
14      "tag": "latest"
15    }
16  ],
17  "_tipi_version": "{{tipi_version_hash}}"
18 }
```



# Zombies in tipi.build !

# Join the open beta



**Tipi.build**

Build

Dependency Management  
+  
Superfast cloud powered C & C++ builds.



**Tipi.run**

Run & CI/CD

Machines sized right for C & C++

Join the Beta !

*The fastest native edit-build-test cycle*

Sign-in now :

**<https://tipi.build/>**

**Enable Promo Code :**

**tipi::build(CPPP)**

3 Skip Ad ►|

*"It could definitely change the whole ecosystem!"*

Arthur Sonzogni – C++ Software Developer

# Join the open beta



**Tipi.build**

Build

Dependency Management  
+  
Superfast cloud powered C & C++ builds.



**Tipi.run**

Run & CI/CD

Machines sized right for C & C++

Join the Beta !

*The fastest native edit-build-test cycle*

Sign-in now :

**<https://tipi.build/>**

**Enable Promo Code :**

**tipi::build(CPPP)**

2 Skip Ad ►|

*"It could definitely change the whole ecosystem!"*

Arthur Sonzogni – C++ Software Developer

# Join the open beta



**Tipi.build**

Build

Dependency Management  
+  
Superfast cloud powered C & C++ builds.



**Tipi.run**

Run & CI/CD

Machines sized right for C & C++

Join the Beta !

*The fastest native edit-build-test cycle*

Sign-in now :

**<https://tipi.build/>**

**Enable Promo Code :**

**tipi::build(CPPP)**

1 Skip Ad ►|

*"It could definitely change the whole ecosystem!"*

Arthur Sonzogni – C++ Software Developer

# Zombies in tipi.build !

```
$> tipi build . -t macos -j 128
```

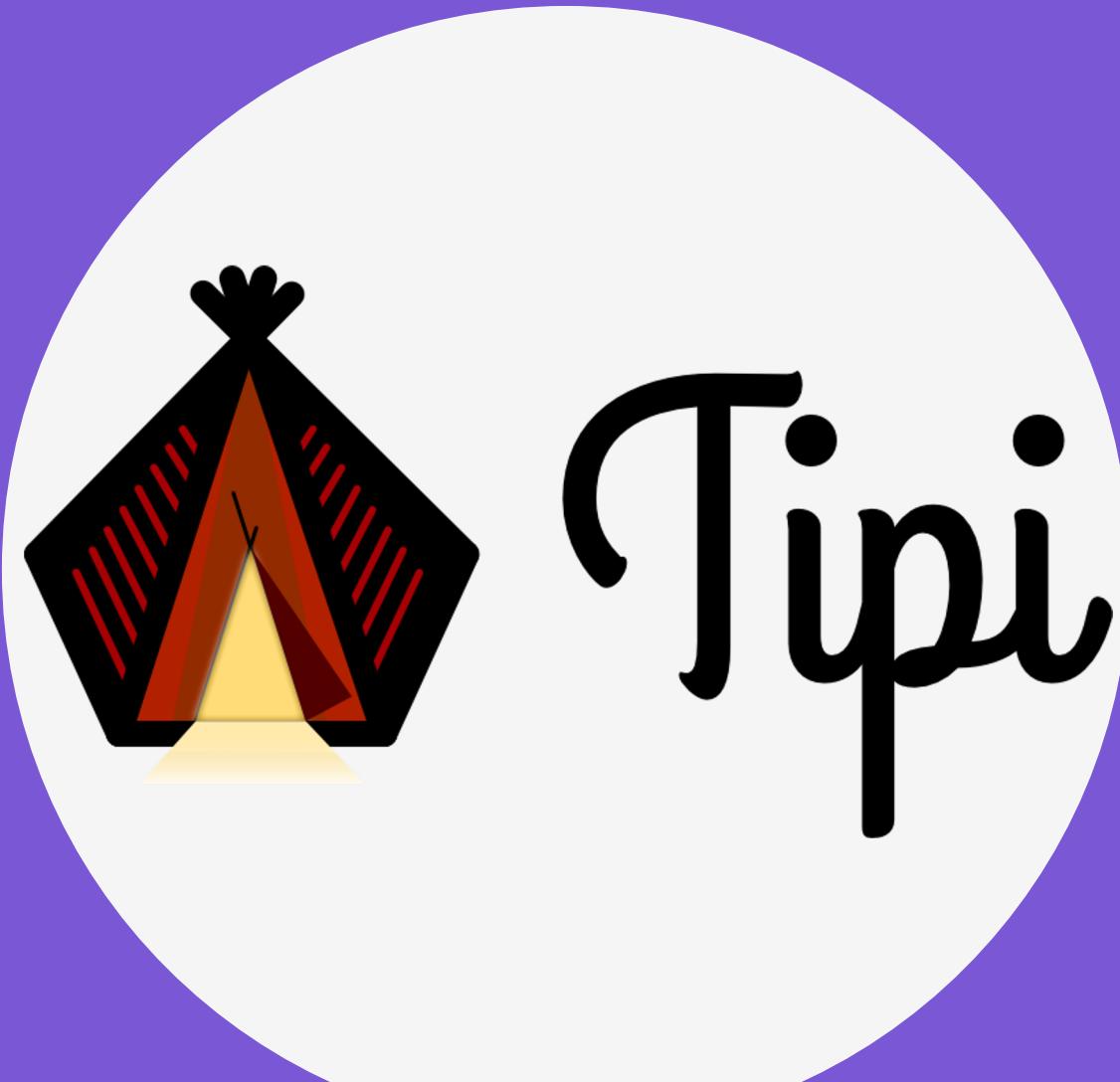
```
 1 namespace tipi::util {
 2
 3     void enable_stacktrace_on_crash() {
 4         std::set_terminate(&detail::on_terminate_print_stacktrace);
 5         ::signal(SIGSEGV, &detail::on_crash);
 6         ::signal(SIGABRT, &detail::on_crash);
 7     }
 8
 9     namespace detail {
10         inline void on_terminate_print_stacktrace() {
11             boost::stacktrace::safe_dump_to("./crash.dump");
12             try {
13                 std::cerr << boost::stacktrace::stacktrace();
14             } catch (...) {}
15             std::abort();
16         }
17
18         inline void on_crash(int signum) {
19             ::signal(signum, SIG_DFL);
20             detail::on_terminate_print_stacktrace();
21             ::raise(SIGABRT);
22         }
23     }
24 }
```

```
$> tipi build . -t macos -j 128
```

```
1 namespace tipi::util {
2
3     void enable_stacktrace_on_crash() {
4         std::set_terminate(&detail::on_terminate_print_stacktrace);
5         ::signal(SIGSEGV, &detail::on_crash);
6         ::signal(SIGABRT, &detail::on_crash);
7     }
8
9     namespace detail {
10        inline void on_terminate_print_stacktrace() {
11            boost::stacktrace::safe_dump_to("./crash.dump");
12            try {
13                std::cerr << boost::stacktrace::stacktrace();
14            } catch (...) {}
15            std::abort();
16        }
17
18        inline void on_crash(int signum) {
19            ::signal(signum, SIG_DFL);
20            detail::on_terminate_print_stacktrace();
21            ::raise(SIGABRT);
22        }
23    }
24 }
```

addr2line  
addr2line

# Thanks



[www.tipi.build](http://www.tipi.build)

Damien Buhl

[damien@tipi.build](mailto:damien@tipi.build)

+41 (0) 78 984 08 13