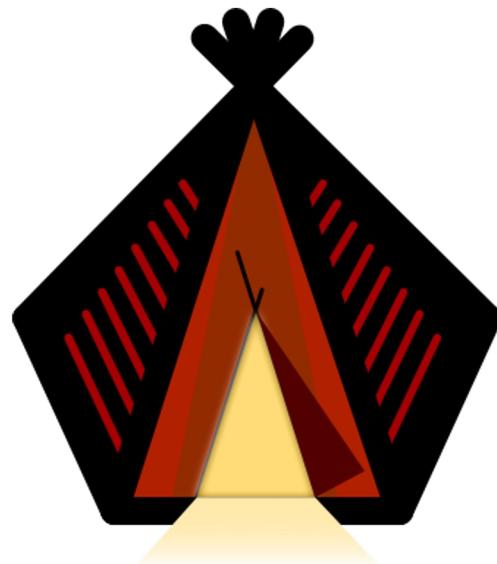


Building a massively scalable remote compiler cloud

Web Enthusiasts Basel – 9th February 2022
@WondrousLTD Basel

www.tipi.build

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



Tipi

Tipi founders



Damien Buhl

CEO & CTO



Yannic Staudt

COO & Business Development



SERPENTINE
Ventures



Advisory Board



Hans Fiesel



JF Bastien



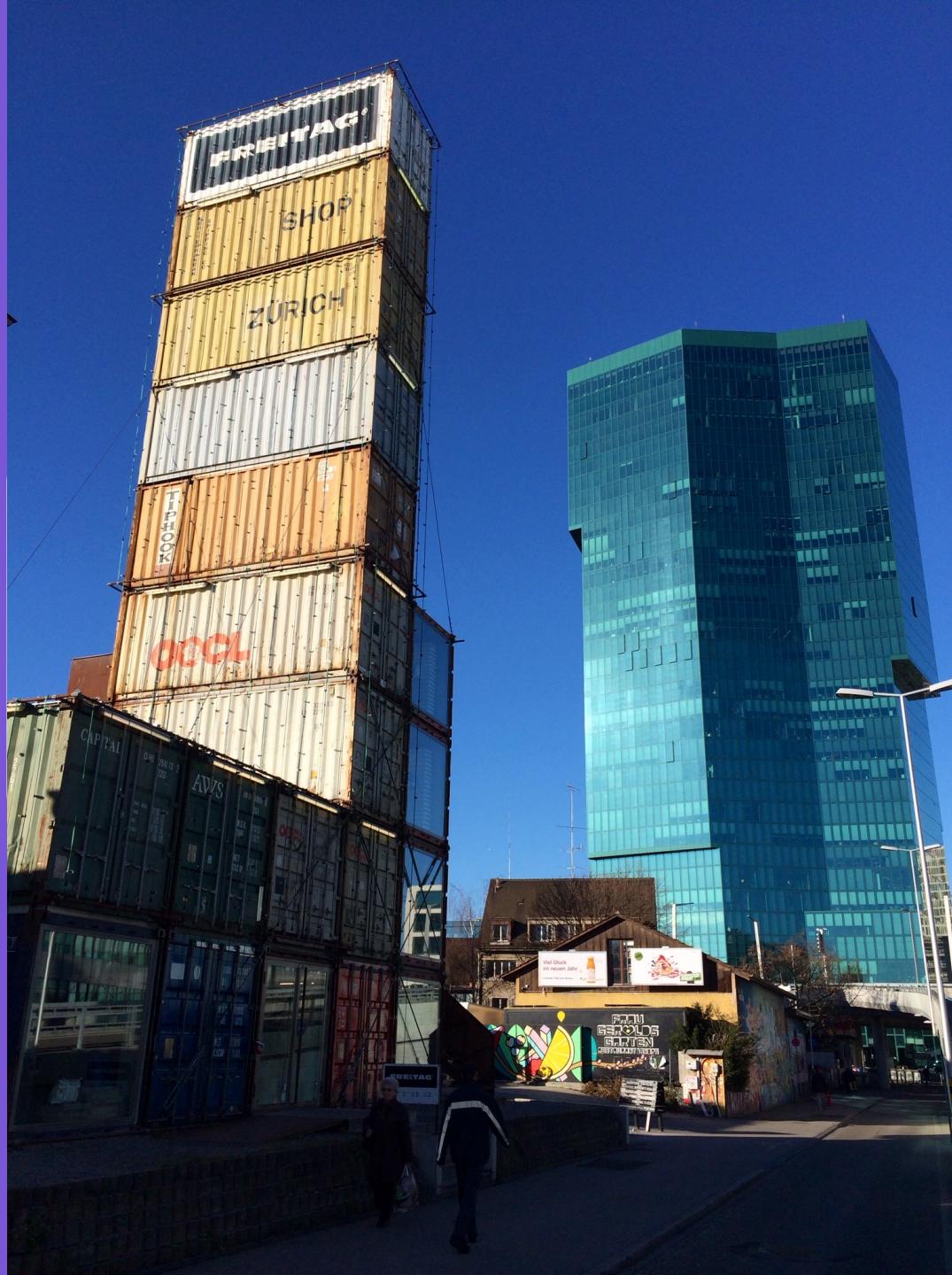
Patrick Wieder



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

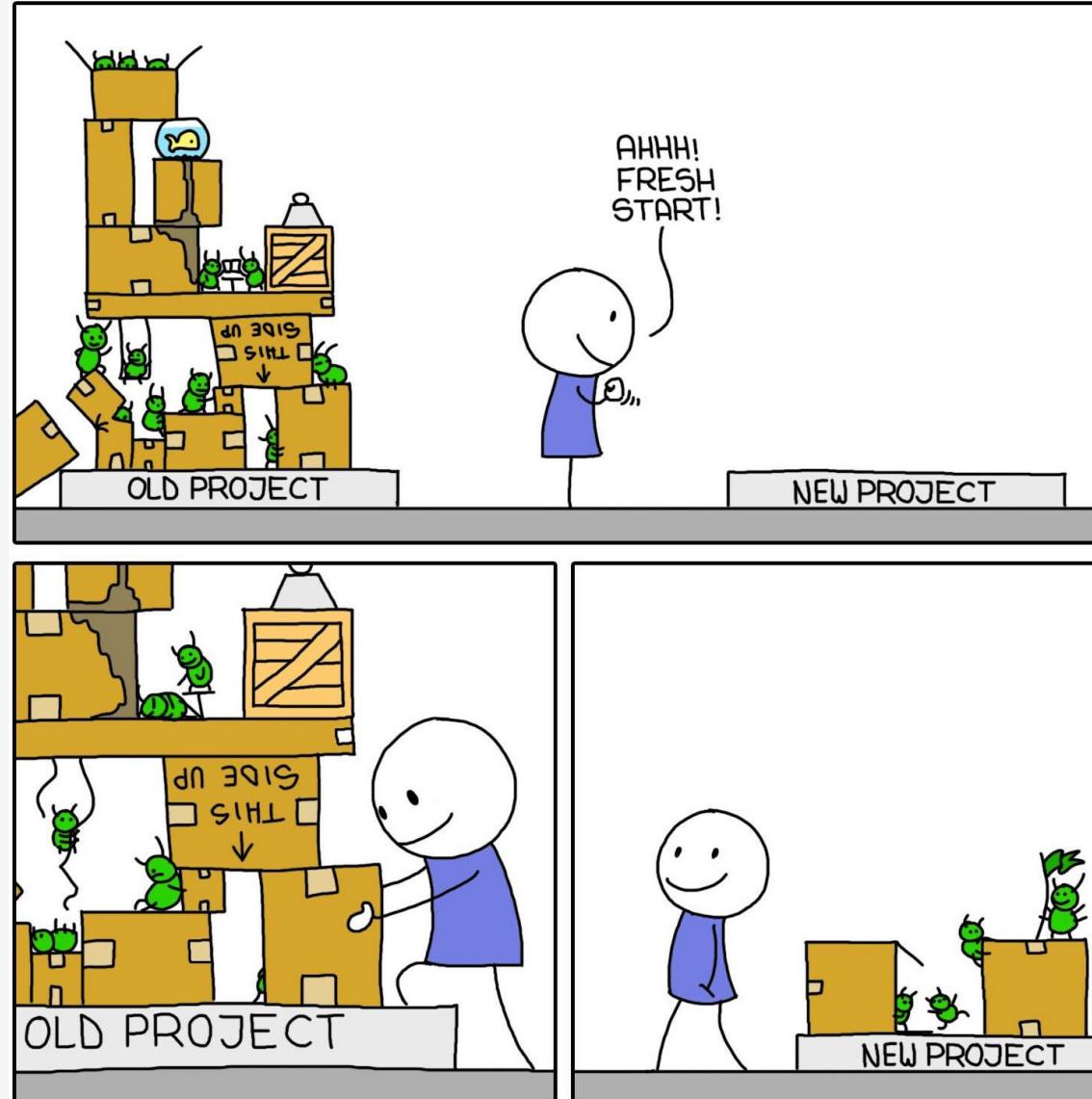
Agenda

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

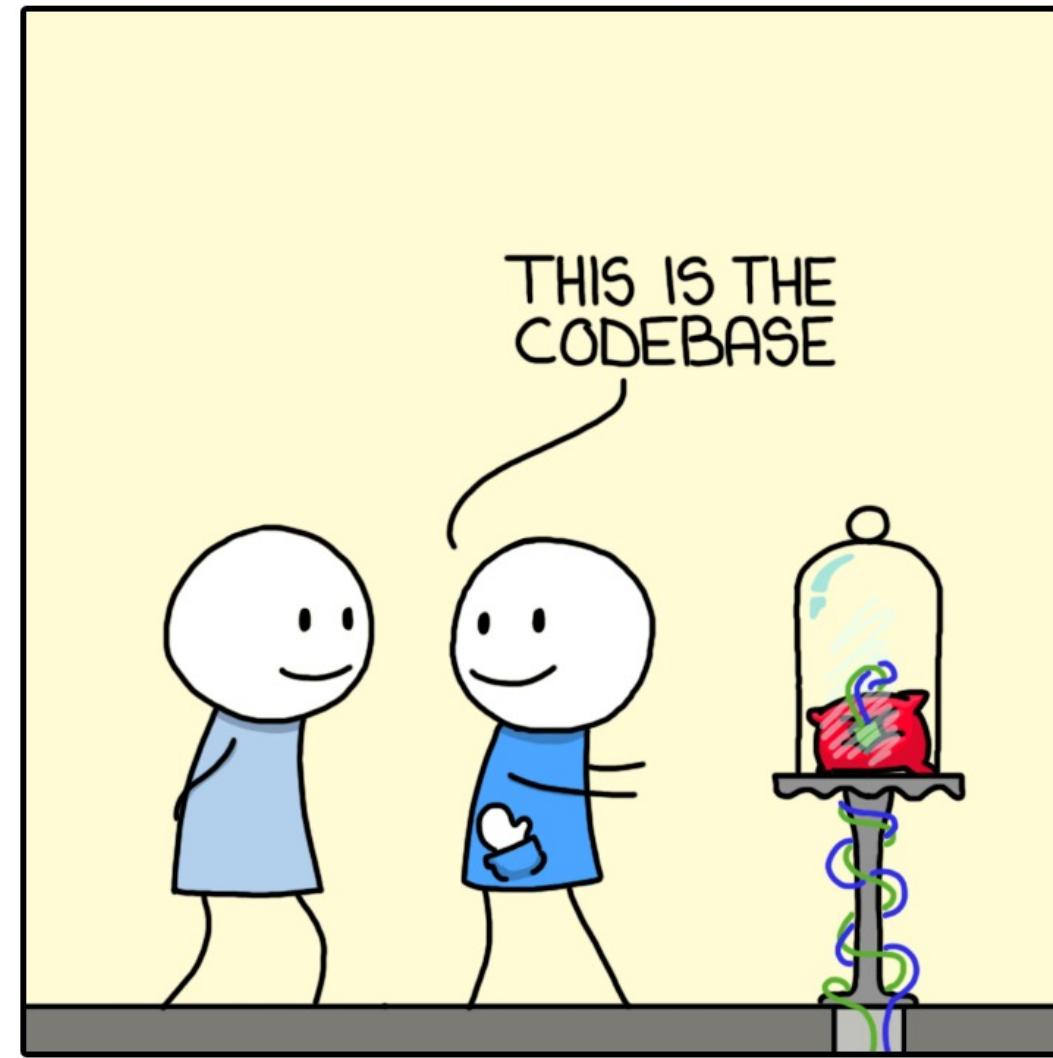


3 problems with C & C++

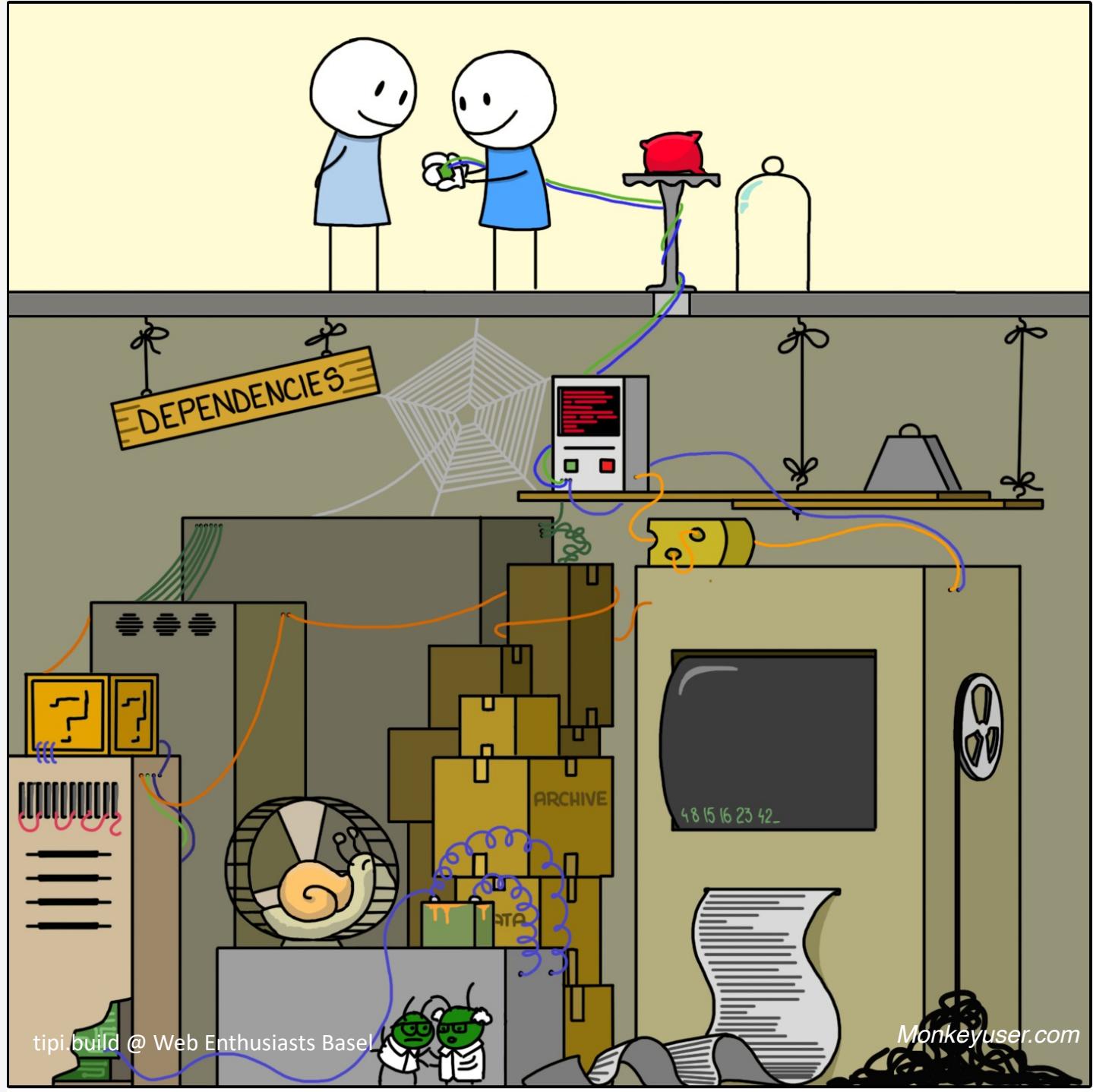
Code Reuse



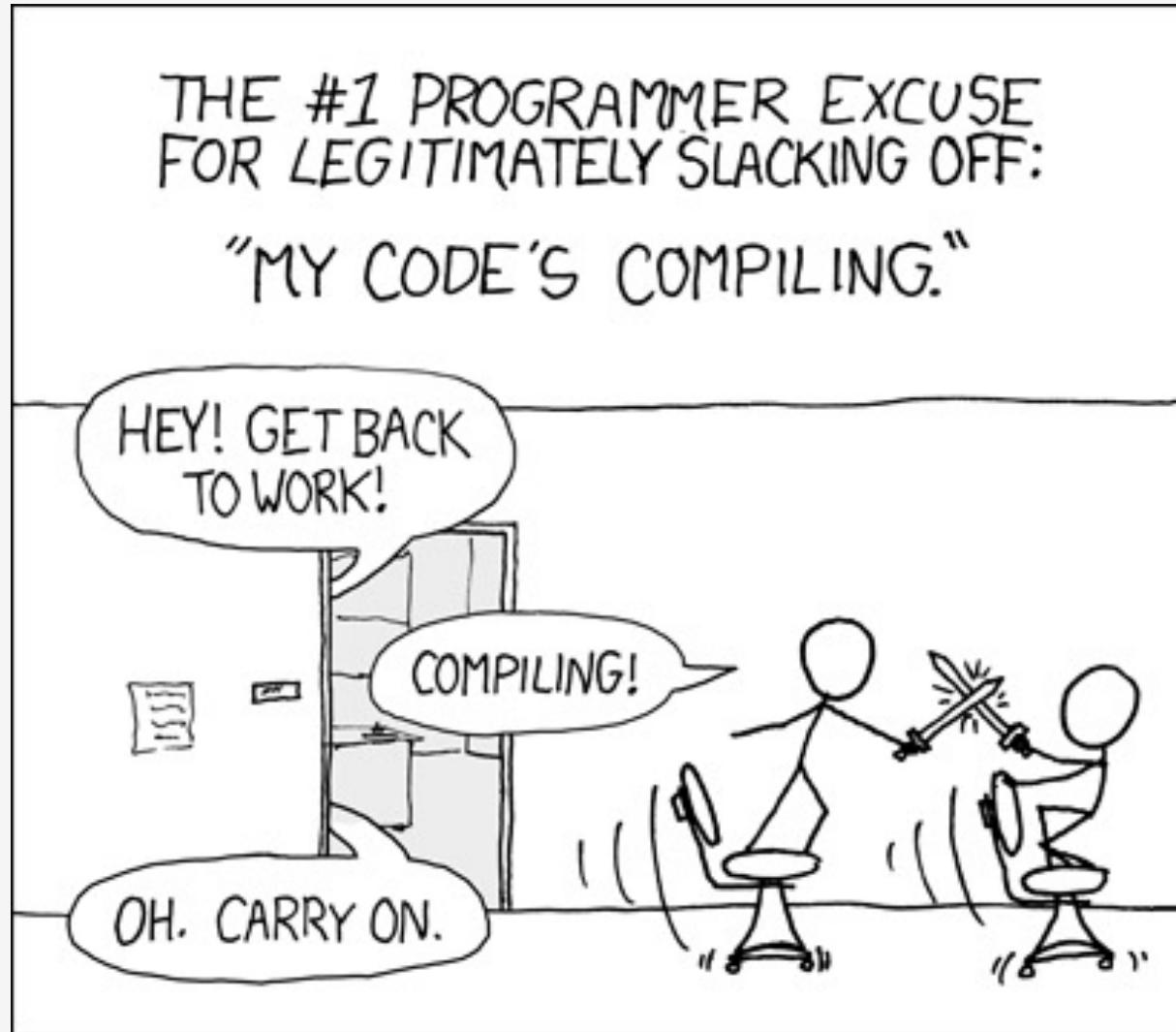
Dependencies & Environments



Comics from Monkeyuser.com



Build & Deploy Time



Comics from xkcd.com

Try tipi ! Give us feedback



Tipi.build

The fastest native edit-build-test cycle for developers

Sign-in now :

<https://tipi.build/>

Enable Promo Code :
THANKS4YOURTIME

Skip Ad ►

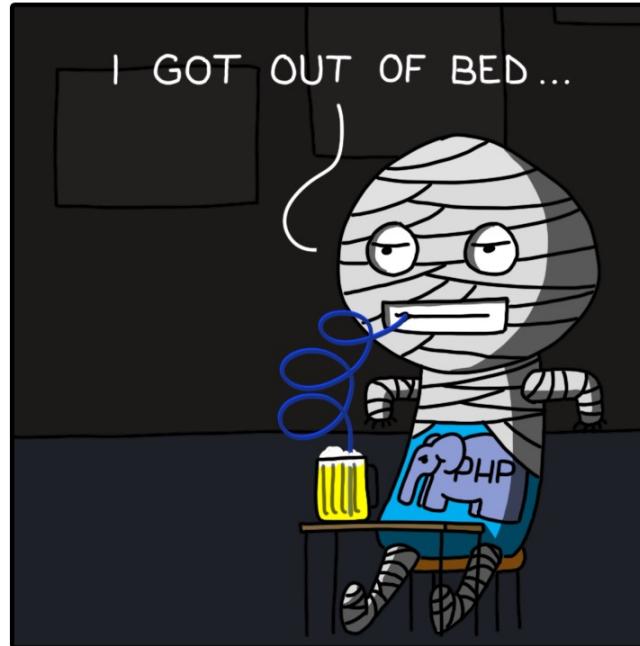
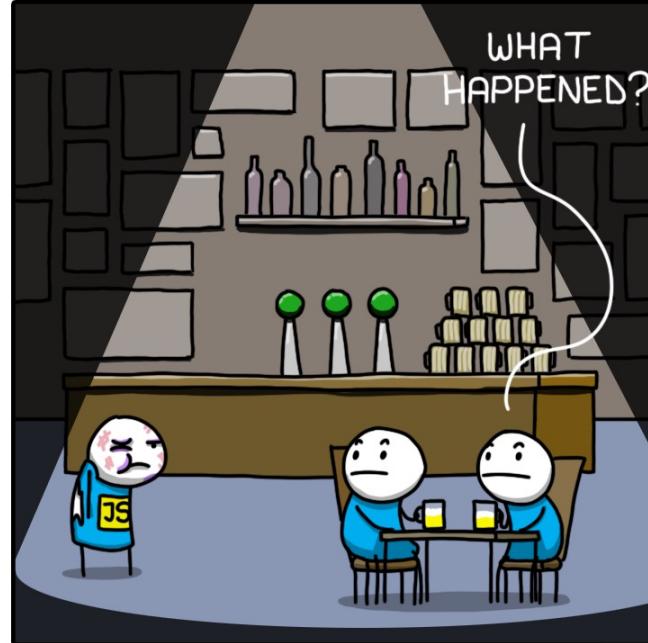
“It could definitely change the whole ecosystem!”

Arthur Sonzogni – Google Chrome Team

Why C++ for webapps ?

MASOCHISM

MONKEYUSER.COM



Seriously ! Why C++ for webapps ?

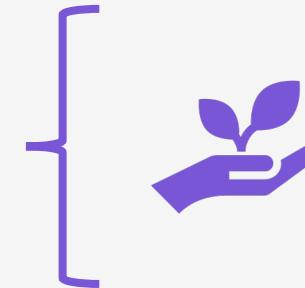
Carbon footprint

Enabling environmentally friendly software



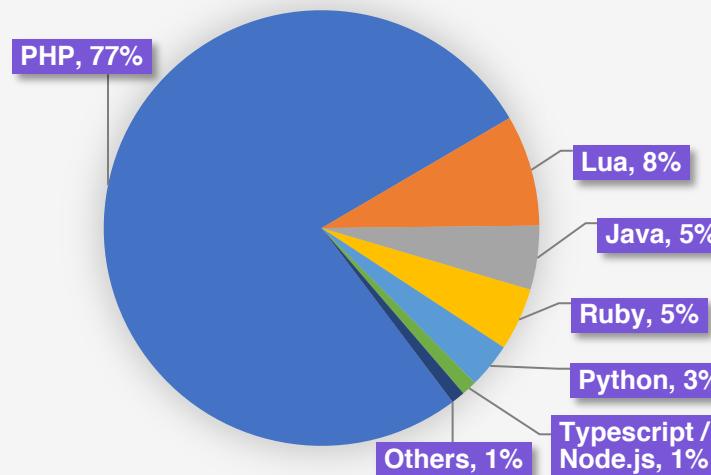
1.32bn CO₂ tons

Yearly Global Emissions by Internet ¹



47'000 CO₂ tons

Yearly Emissions addressable ² with C++



186M

Of all internet requests by seconds served by non-native technologies globally ³

¹ The Shift Project, Lean ICT, March 2019

² Cisco Annual Internet Report 2020 + HTTP Archive, State of The Web 11.2020

³ 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 :



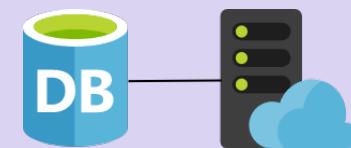
Dell R440 Xeon Gold
1480 kWh Energy Demand / year

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



12'880 kgCO₂e / year

C++



1'840 kgCO₂e / year

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

First part of our journey : Microservices vs Monoliths

Microservices vs Monoliths

Monoliths

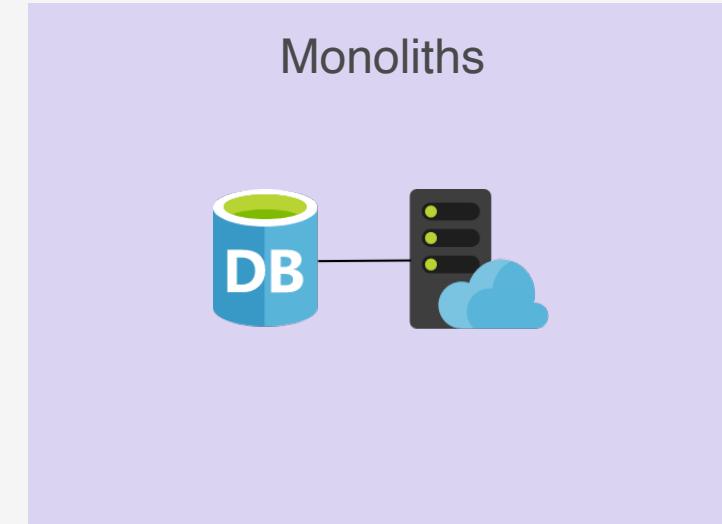
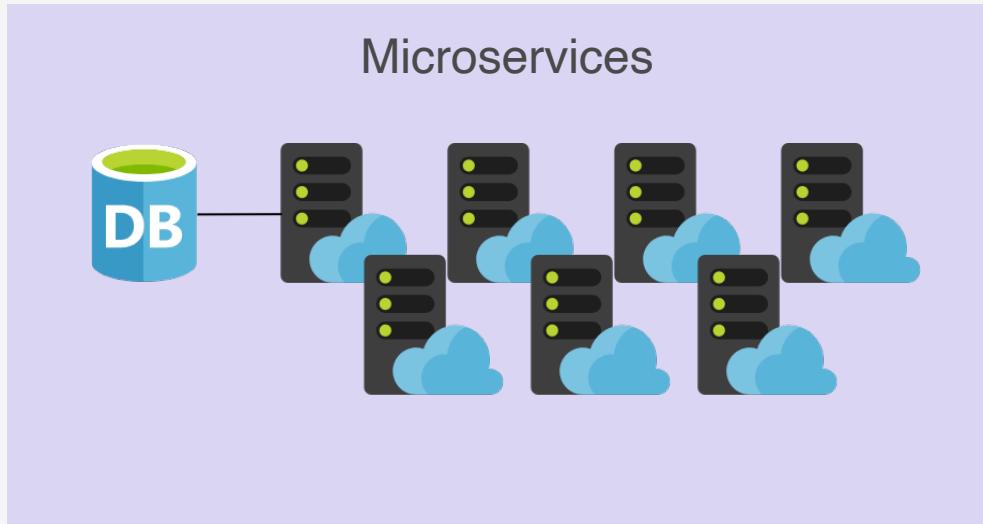
- Vertical scalability only
- Availability issues
- + Simplicity
- + Everything in the same memory space

Microservices vs Monoliths

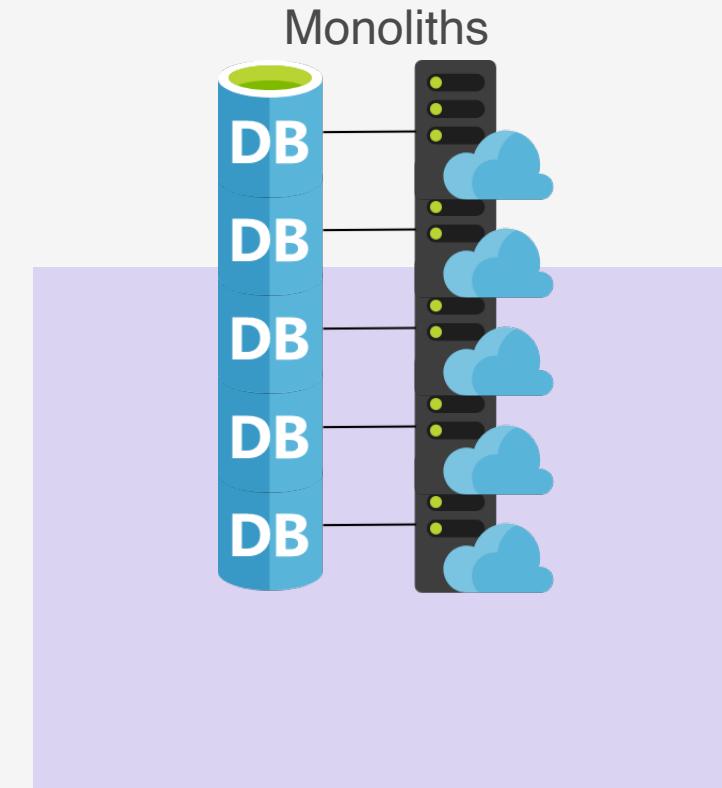
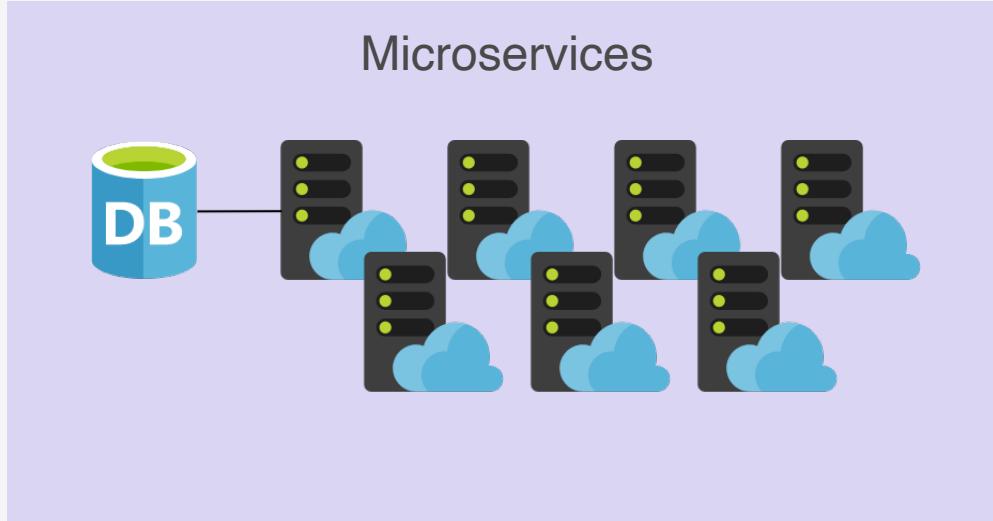
Microservices

- Lot of Marshalling code between services
 - Shared state data races
- ~ Prone to Conway's Law
- + Horizontal scalability
 - + High Availability
 - + Degraded operation mode
 - +? Forces to design the App in a scalable way

< 1000 users



> 1000 users

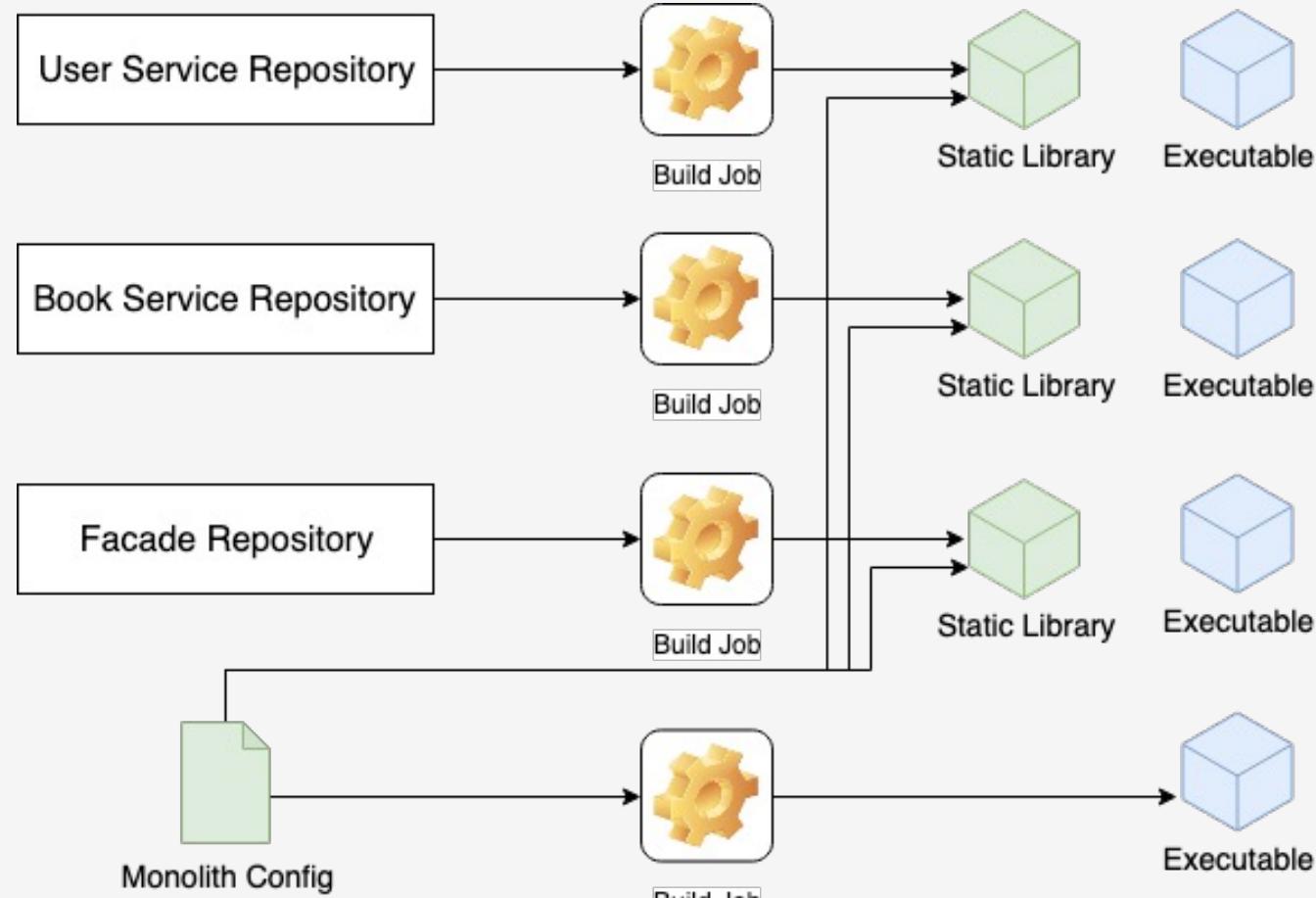


Monolithization

Term coined by Leonid Stryzhevskyi (@lganzzzo)

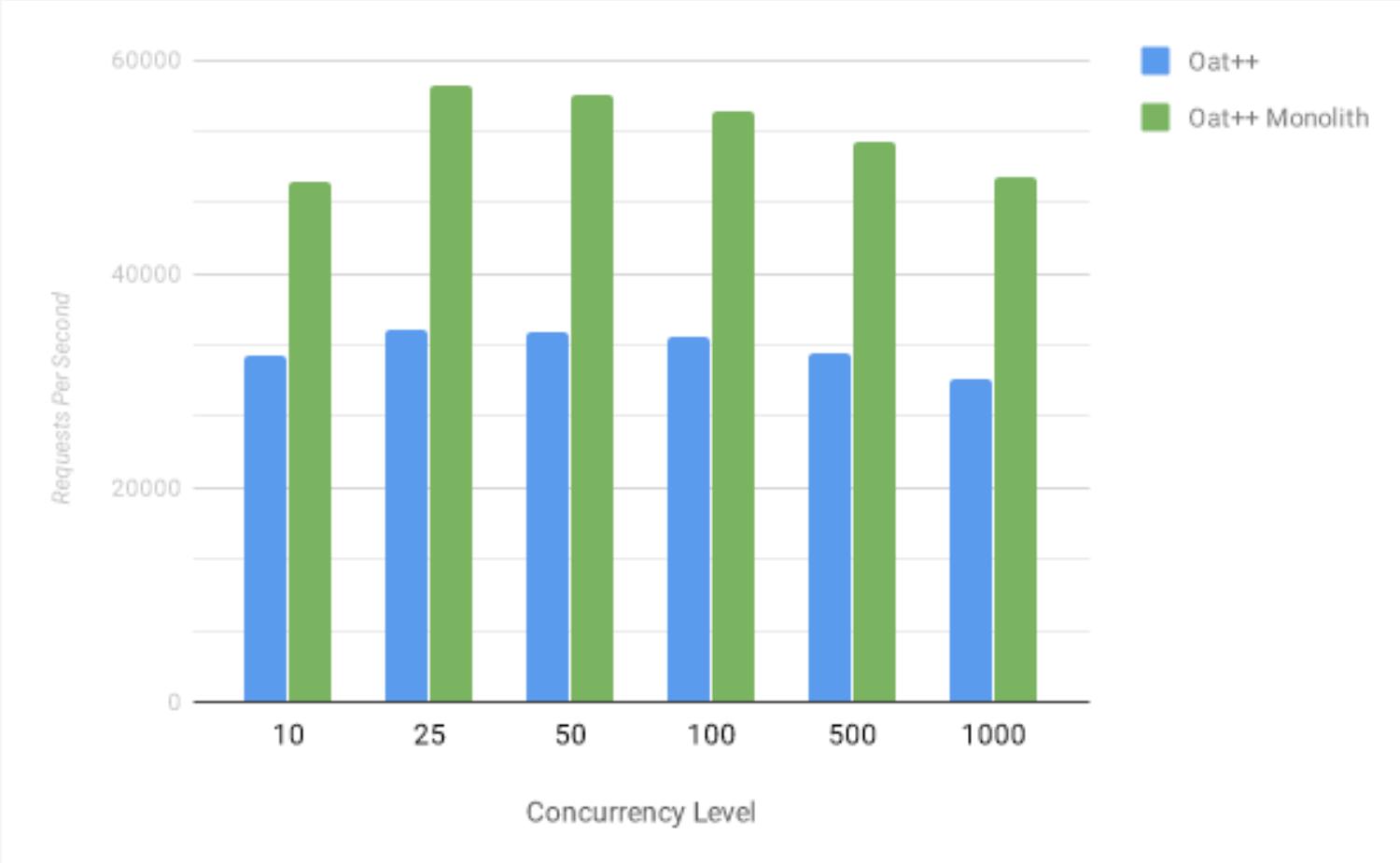
<https://oatpp.io/docs/monolithization/>

Monolithization



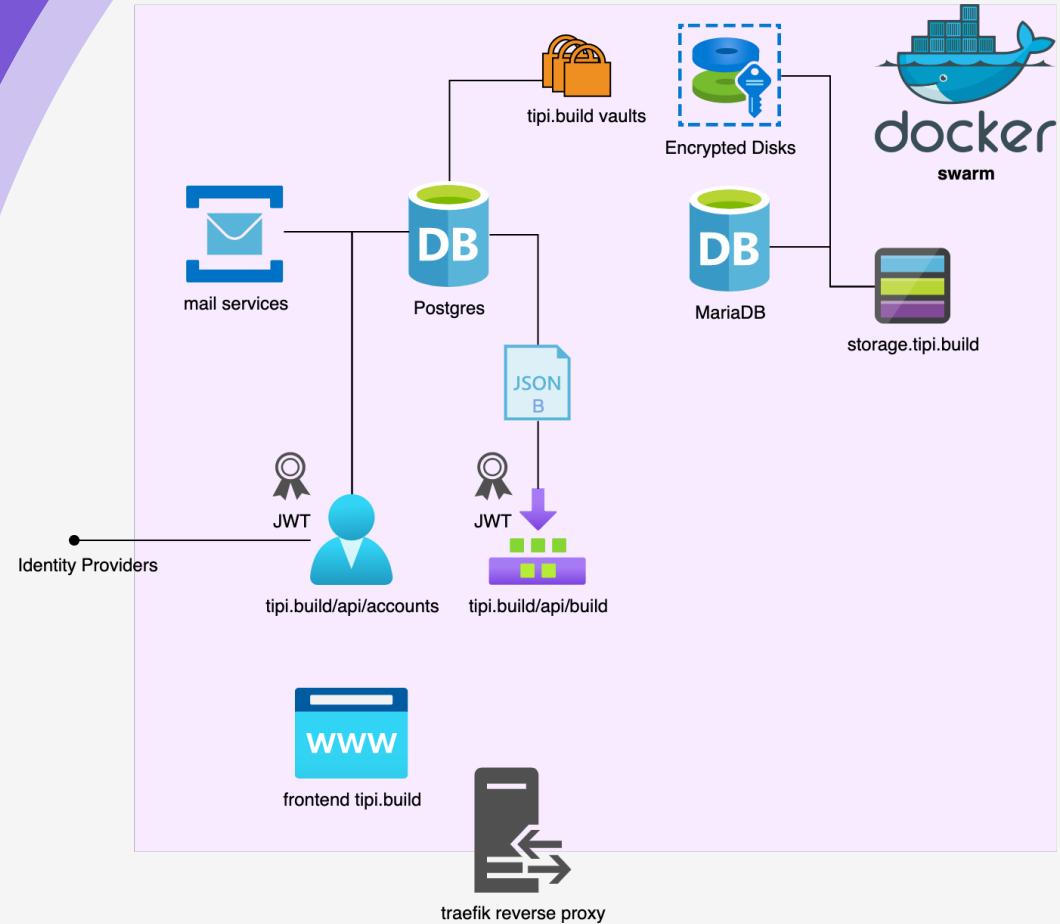
Source : oatpp.io

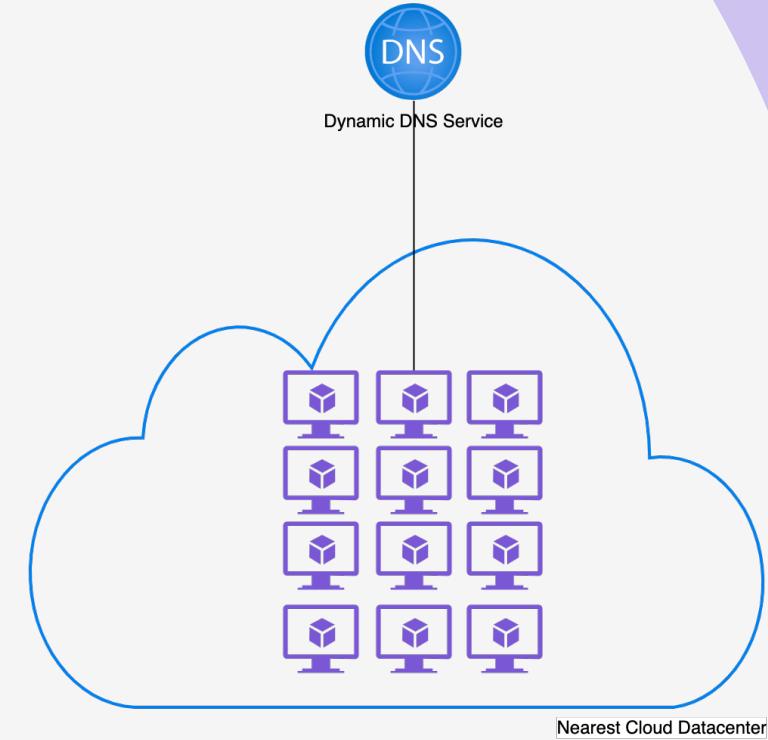
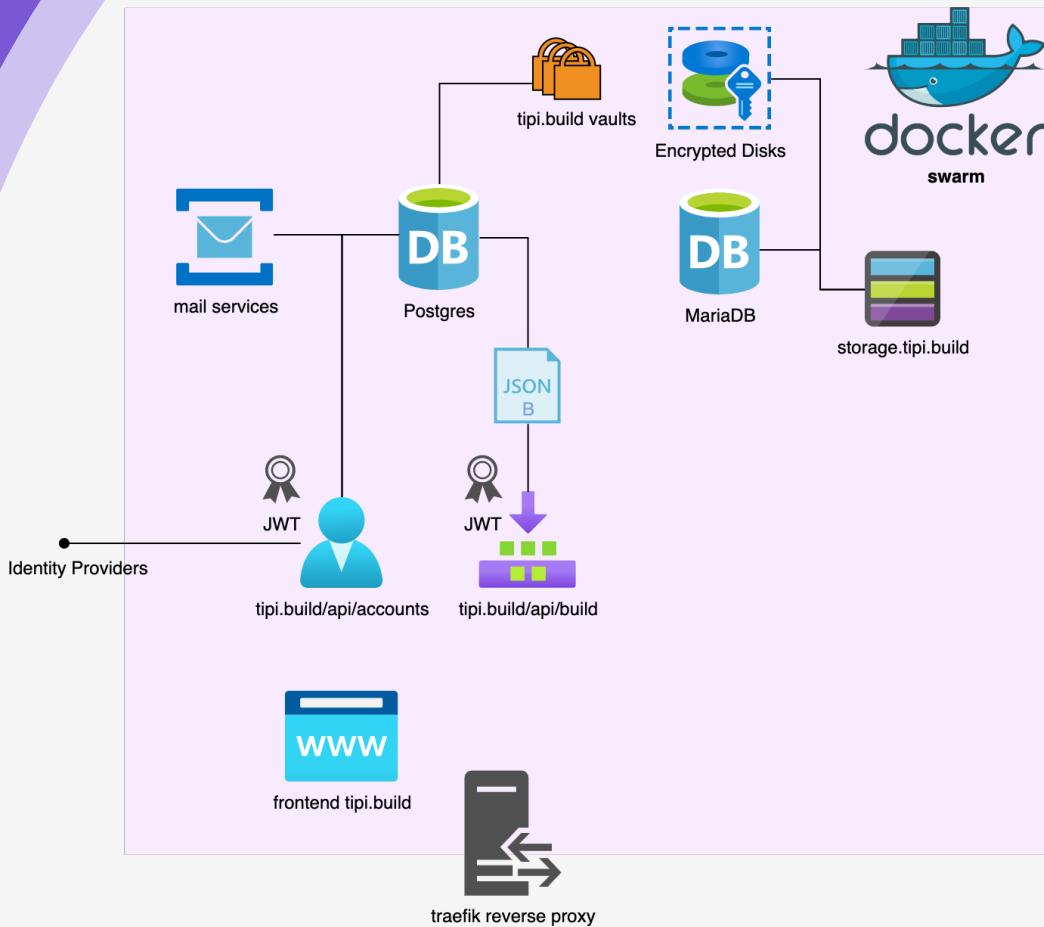
Monolithization

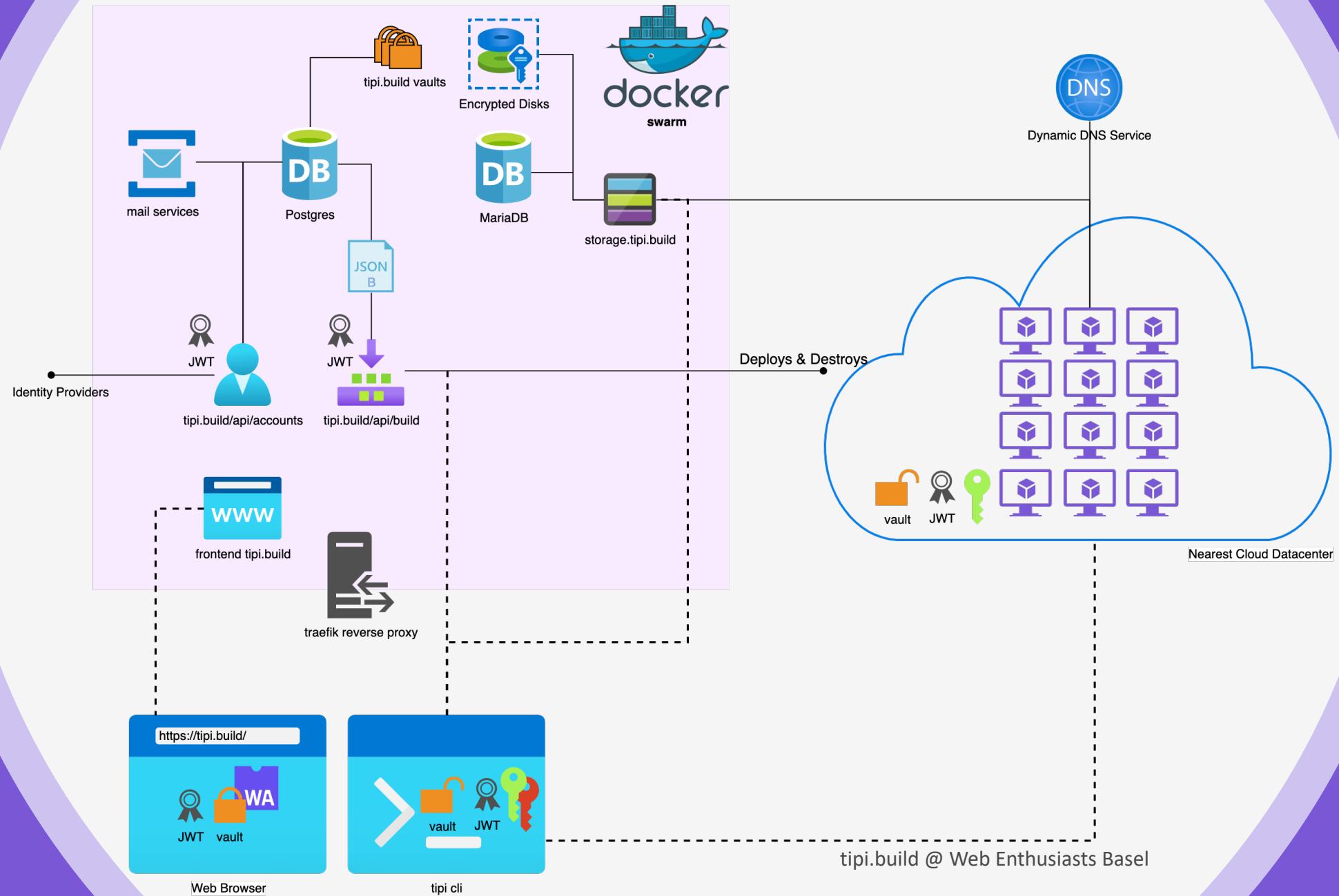


Source : oatpp.io

How is **Tipi.build** built ?







How is **Tipi.build** built ?



Tipi.build

is
built with



Tipi.build

C++

Technology Stack





Tipi.build helps reusing code.
Reuse code !



We dogfood but...



Tipi.build needs you !

<https://tipi.build/>

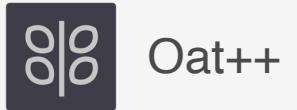
Enable Promo Code :
THANKS4YOURTIME

Be part of the C++ build revolution !

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

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



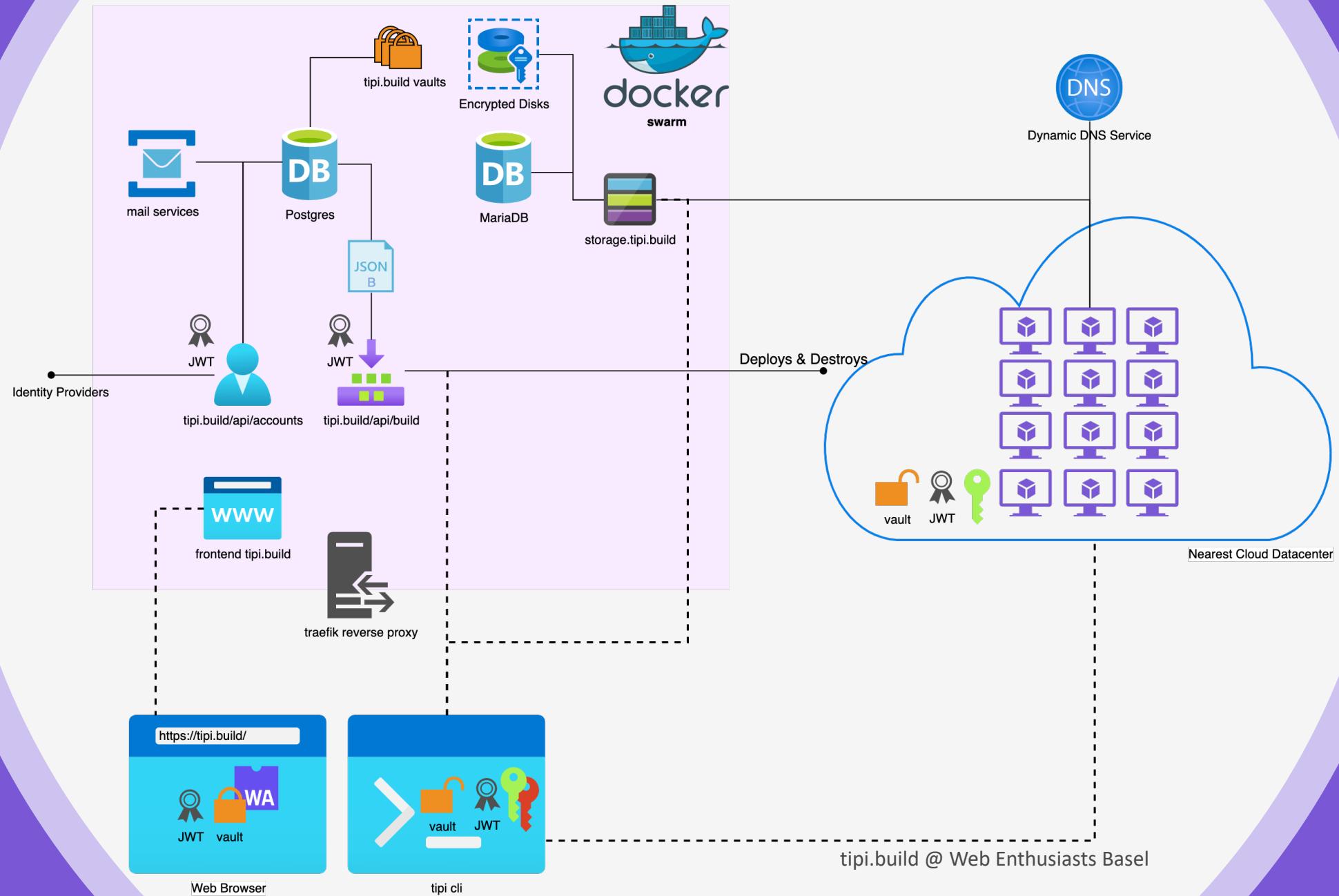
docker

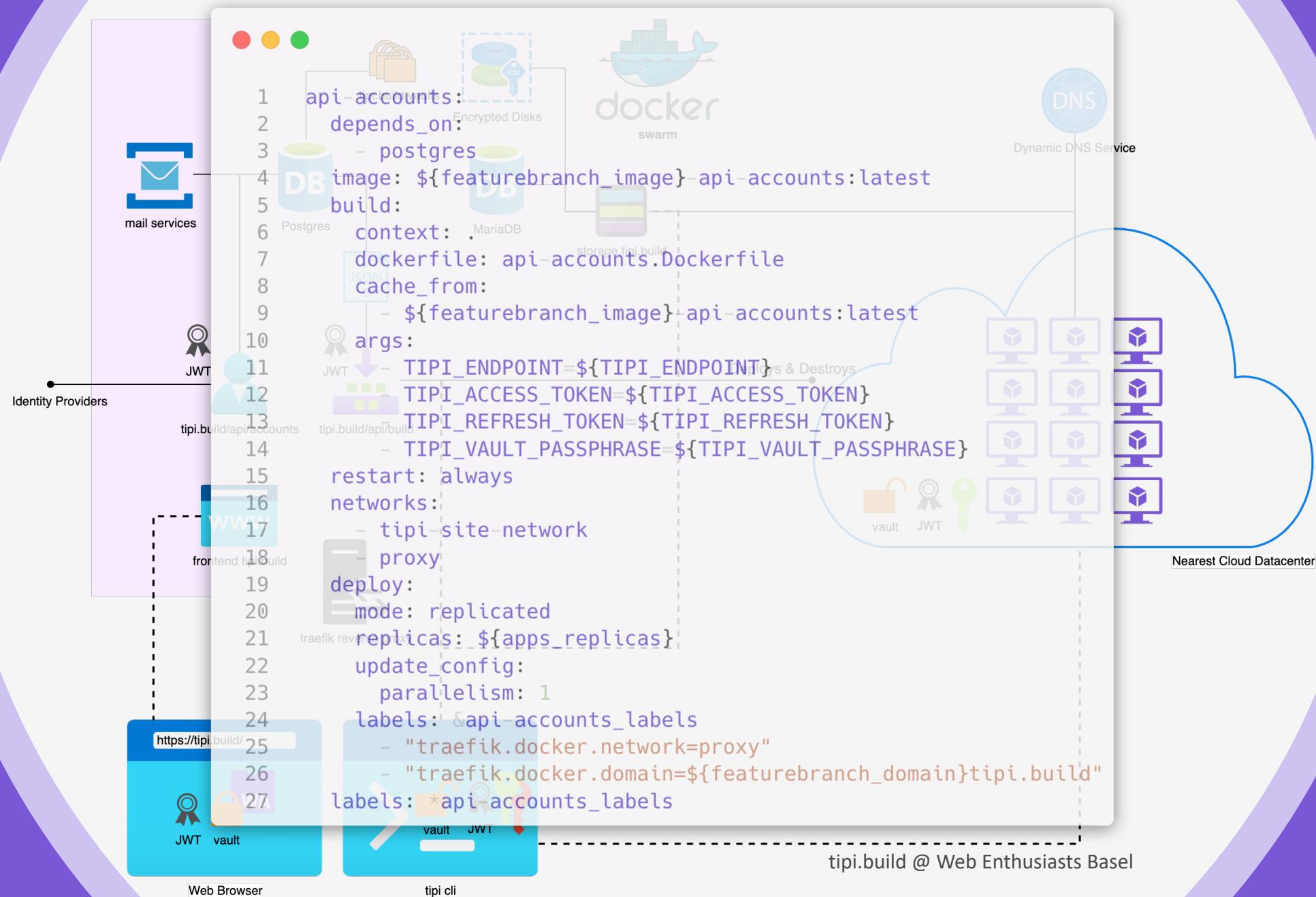


Tipi.build

Why Docker Swarms ?







OpenAPI REST in C++



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 Server

Oat++ OpenAPI Server

```
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

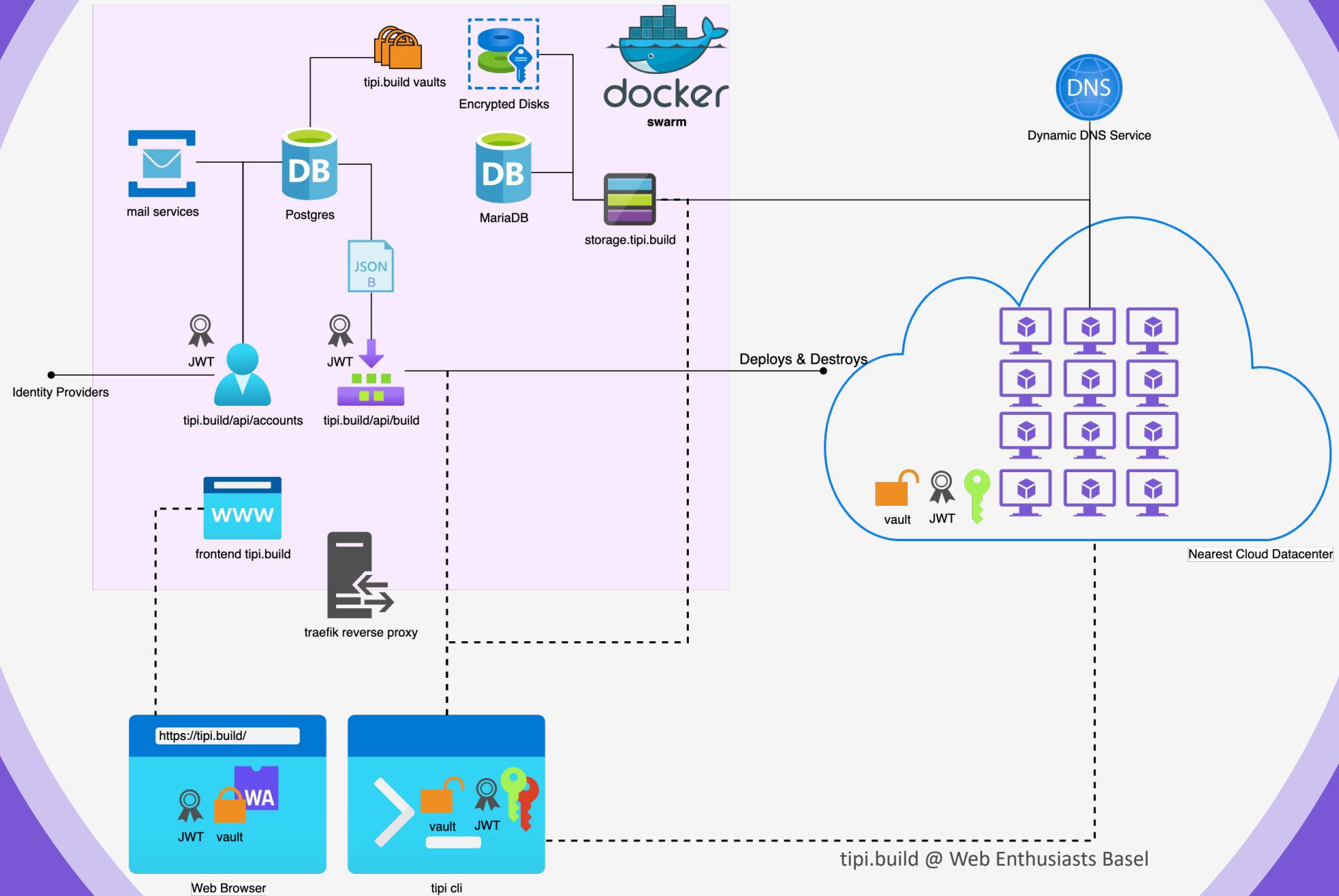
[Our Adventures With REST API in C++ : Making it Easy](#)

OpenAPI REST in JS



Case Study from developers PCs to the tipi.build cloud, via REST

tipi.build vault





explore NEW documentation more ▾

Dashboard

@daminetreg ▾

Manage your vaults

 Lock your vaults

daminetreg's vault

private 

Your access rights: admin

Access tokens:



github.com
User: daminetreg

[Renew token](#)

[Details on Github](#)



tipi.build secure storage



github.com
User: Banana



[+ Add credentials](#)

[Change my passphrase](#)



C++ Vault in Webassembly with JS Bindings

```
 1 class vault {
 2     public:
 3
 4     //! \brief Loads a vault with the given key from the provided encrypted_buffer.
 5     vault(const vault_access_key& access_key, const std::string& encrypted_buffer);
 6
 7     //! \brief Adds the provided auth to the vault
 8     void add(const auth_t& auth);
 9
10    std::string get_encrypted_buffer() const;
11 };
12
13 EMSCRIPTEN_BINDINGS(vault_bindings) {
14     class<vault>("tipi_vault")
15         .constructor<const vault_access_key&, std::string>()
16         .function("add", &vault::add)
17         .property("encrypted_buffer", &vault::get_encrypted_buffer, &vault::set_encrypted_buffer)
18     ;
19 }
```

Updating a vault entry

In Nuxt.js vuex store

```
1 // Call into Webassembly
2 const vault = new tipi.tipi_vault(passphrase, data);
3 vault.add(entry)
4
5 // Get Vault Encrypted Data from Wasm
6 const newEncryptedData = vault.encrypted_buffer;
7
8 // Call OpenAPI Rest
9 await apiAccount.update_vault(uuid, {
10   vaultData: newEncryptedData,
11   vaultName: storageVault.vaultName
12 });
```

OpenAPI Client JS

tipi web OpenAPI Client Generator

npm v22.3.21-d downloads 3.6k

Credit where credit is due: this library is largely based on the awesome work of @mikestead with the awesome `openapi-client` library. We decided to give it an overhaul to cope with all the "new" goodness!

Generate ES6 client integration code from an OpenAPI 3 spec.

Also supports optional Redux action creator generation.



Tested against JSON services built at tipi.build and the official [OpenAPIv3 petstore example](#)

Install

In your project

```
> npm i @pysco68/tipi-web-openapi-client
```

```
287 /**
288 *
289 * @typedef update_vault_request
290 *
291 * @property {string} vaultName
292 * @property {string} vaultData <byte> Vault encrypted using the key in accessKey
293 */
294
295 /**
296 *
297 * @typedef update_vault_response
298 *
299 * @property {string} vaultId <uuid>
300 * @property {string} vaultETag Updated ETag of the vault
301 */
302
365 /**
366 * Update a single vault the user has access by UUID
367 *
368 * @param {string} uuid
369 * @param {object} options Optional options
370 * @param {_requestOptionsType} [options.requestOptions] Per-call request options
371 * @param {update_vault_request} body
372 * @return {Promise<$tipi$ApiResponse<update_vault_response>>}
373 */
374 export function update_vault(uuid, body, options) {
```

OpenAPI Server

express-joi-openapi

```
1 export const expressOpenApi = new ExpressOpenAPI();
2
3 expressOpenApi.specification.setComponent('securitySchemes',
4 'BearerAuth', {
5   type: 'http',
6   scheme: 'bearer',
7 });
8
9 const specification = expressOpenApi.populateSpecification(app)
10 specification.addServer({ url: ENDPOINT });
11
12 app.use('/api-docs',
13   swaggerUi.serveWithOptions(),
14   swaggerUi.setup(specification));
```

```
1 /**
2  * PUT /api/accounts/v1/vaults/{UUID}
3 */
4 api.put('/:uuid',
5   operationId('update_vault', 'Update a single vault the user has access by UUID'),
6   validateRequest({
7     params: {
8       uuid: Joi.string().guid().required()
9     },
10    body: Joi.object({
11      vaultName: Joi.string().max(255).required(),
12      vaultData: Joi.string().base64().description('Vault encrypted using the key in accessKey')
13    })
14  }),
15  validateResponse({
16    200: { body:
17      Joi.object({ vaultId: Joi.string().guid().required() })
18    }
19  }),
20  ensureAuthenticated(),
21  asyncHandler(async (req, res) => {
22    // ...
23    // Express JS
24    // ...
```



Tipi.build deploys many
manycores machines !

tipi.build environments



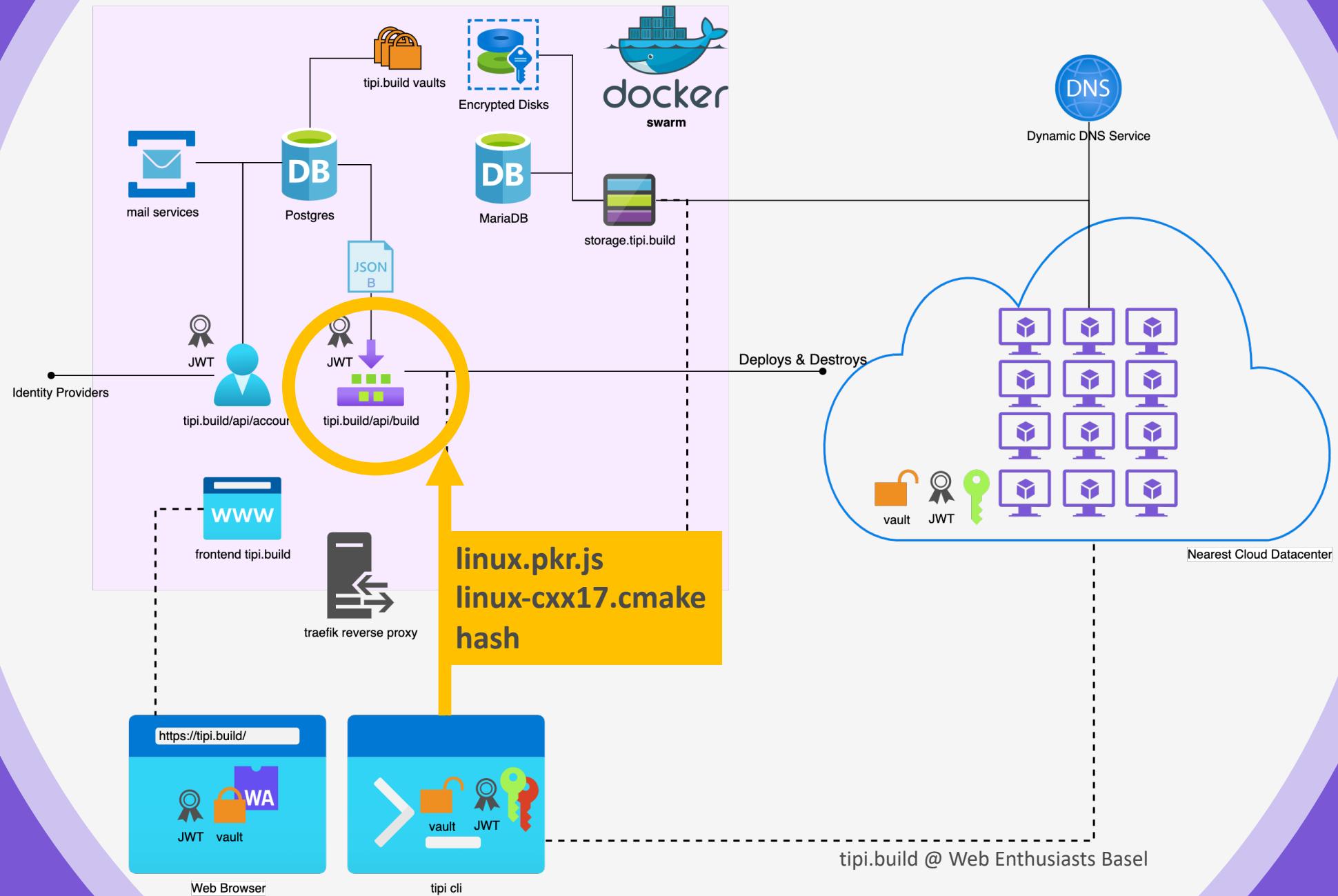
linux-cxx17.cmake

```
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 !

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

Damien Buhl

damien@tipi.build

+41 (0) 78 984 08 13



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

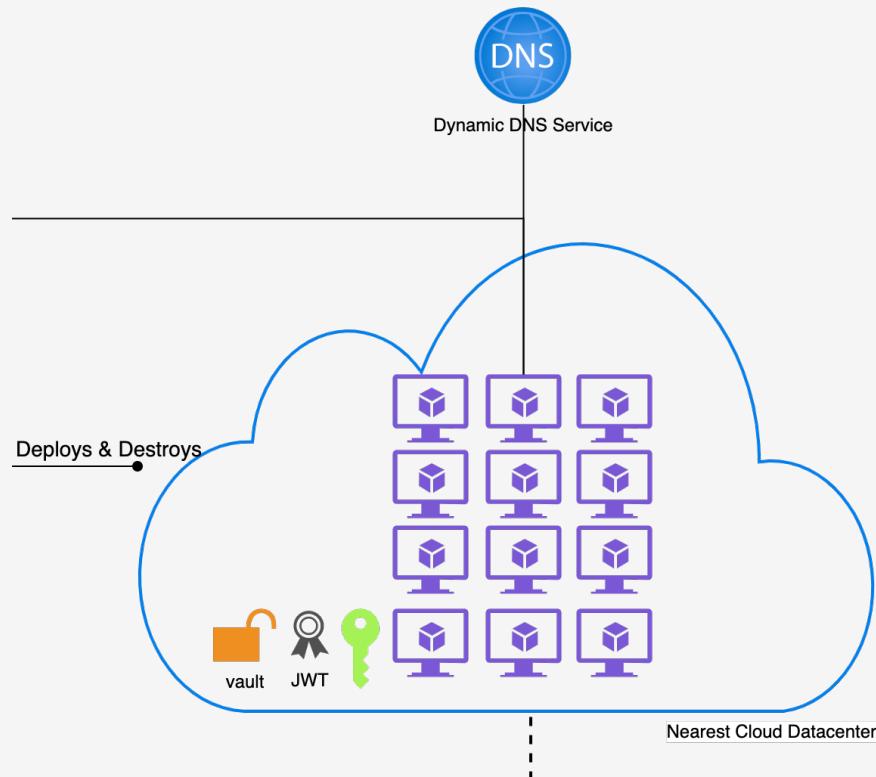
ORM

*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 ?