Pizza +  = <3

En kort historie om pizza, gambling og rust

Hvem er jeg?

Mats Jun Larsen
NTNU / KodeWorks / Dotkom

Ok, så hva er greia?

Bak F.V: Mads Bårnes, Brage Baugerød
Foran F.V: Herman Sætre, Nora Langli, Sondre Alfnes, Julian Grande, Mats Larsen
Ikke til stede: Thomas Hasvold, Anhkha Vo, Njål Sørland, Isak Solheim, Hanna Lunne, Jo Tjernshaugen, Billy Steen Barret

Sultne code monkeys

Løsningen?

Henrik Skog 15:57

/spin

PizzaPicker APP 15:57

Gratulerer, du har fått 21 MR.X

21 MR.X er en pizza med Ost, tomatsaus, kjøttde
hvete, melk, selleri)

Litt inspirasjon

Blank AS' innlegg

Blank AS
2 450 følgere
6md

"Hadde vært gøy å få ukentlig random nettside fra random tid en gang i uka i **#random** kanalen, for å mimre litt og se hvordan ting har utviklet seg. Mulig å enkelt hente ut det av wayback machine?" — Martin, Designer i Blank

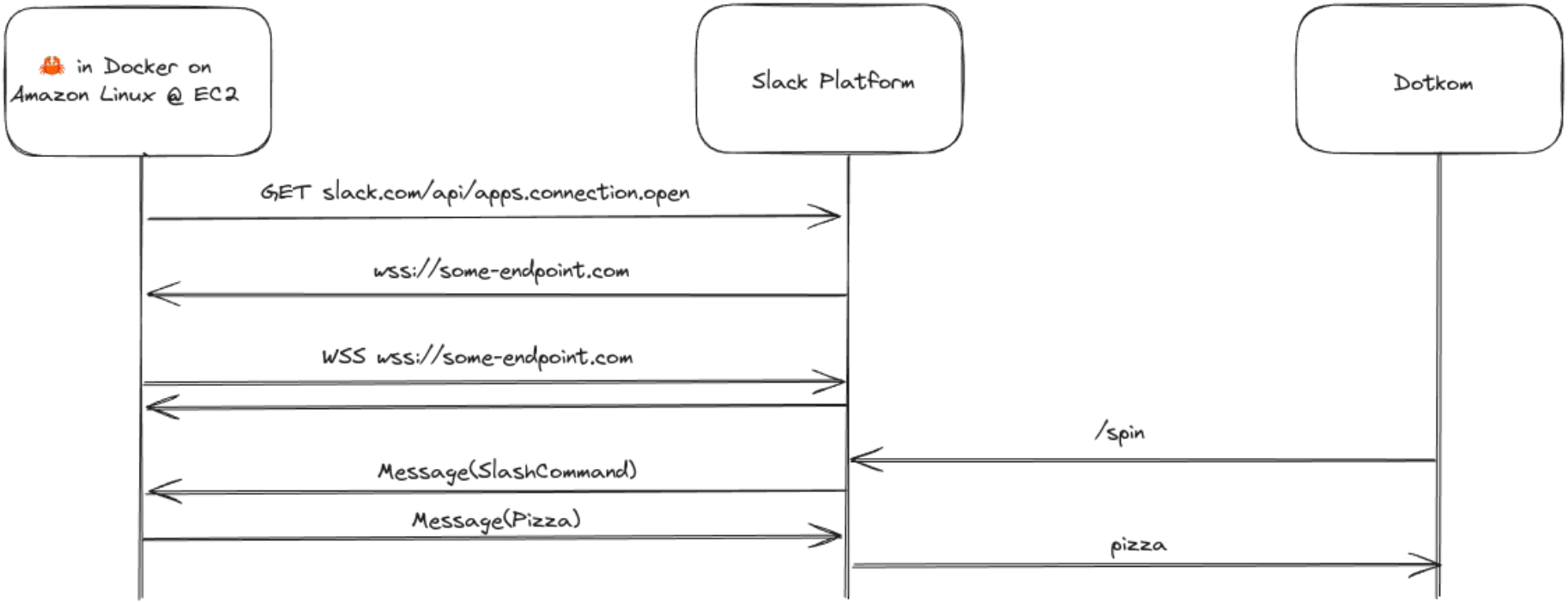
Theodor René Carlsen tok denne ideen fra en intern slacktråd og gjorde den til virkelighet. Sjekk ut blogginnlegget hvor han skriver om implementasjonen og hvilke fartsdumper han møtte på veien 😄 🦀

Slack + Rust?

- https://github.com/slack-rs/slack-rs er ikke maintained :(
- https://github.com/abdolence/slack-morphism-rust ser lovende ut :O

Men hva om vi hjemmesnekrer alt? :D

Hvordan funker det?



Stacken

- **Tokio**: Async Runtime
- **Reqwest:** Async non-blocking HTTP requests
- **Tracing**: Async-aware tracing og logging

- **Tungstenite**: WebSocket Server + Client for 🦀
- **Serde**: type-safe JSON meldinger fra og til Slack

Få websocket endepunkt

```rust
#[tracing::instrument]
pub async fn get_websocket_endpoint(client: &Client) -> Result<String> {
    tracing::info!("Requesting websocket endpoint from Slack");
    let response = client
        .post("https://slack.com/api/apps.connections.open")
        .send()
        .await?
        .json::<serde_json::Value>()
        .await?;
    let url = response["url"].as_str().expect("url not found in response");
    tracing::info!("Received websocket endpoint: {}", url);
    Ok(url.to_string())
}
```

Bruk endepunktet

```rust
#[tracing::instrument]
fn establish_websocket_connection(url: &str) -> SlackWebSocket {
    let (socket, response) = tungstenite::connect(url).expect("Failed to connect to websocket");
    tracing::info!(
        "Request for websocket connection returned {}",
        response.status()
    );
    socket
}

type SlackWebSocket = WebSocket<MaybeTlsStream<std::net::TcpStream>>;
```

Lytte til WebSocket Events

```
loop {
    let msg = socket.read().expect("Failed to read from websocket");
    // The Slack API promises that messages are sent as JSON, so we can safely assume that
    // the message is a JSON string
    let msg = match msg {
        Message::Text(msg) => msg,
        Message::Ping(_) => {
            tracing::debug!("Received ping from Slack websocket");
            socket
                .send(Message::Pong(vec![]))
                .expect("Failed to send pong");
            continue;
        }
        _ => {
            tracing::warn!("Received non-Text message from Slack websocket");
            continue;
```

Og med Serde blir alt typesikkert :D

```
let json = serde_json::from_str::<incoming::SlackIncomingMessage>(&msg);
if let Err(e) = json {
    tracing::warn!(
        "Failed to parse JSON message from Slack: {} from JSON {}",
        e,
        msg
    );
    continue;
}
#[derive(Deserialize, Debug)]
#[serde(tag = "type")]
pub enum SlackIncomingMessage {
    #[serde(alias = "slash_commands")]
    SlashCommands(Box<Incoming<SlashCommandIncomingMessage>>),
    #[serde(alias = "disconnect")]
    Disconnect(Box<SlackDisconnectIncomingMessage>),
    #[serde(alias = "hello")]
    Hello(Box<SlackHelloIncomingMessage>),
```

Håndtere gambling :)

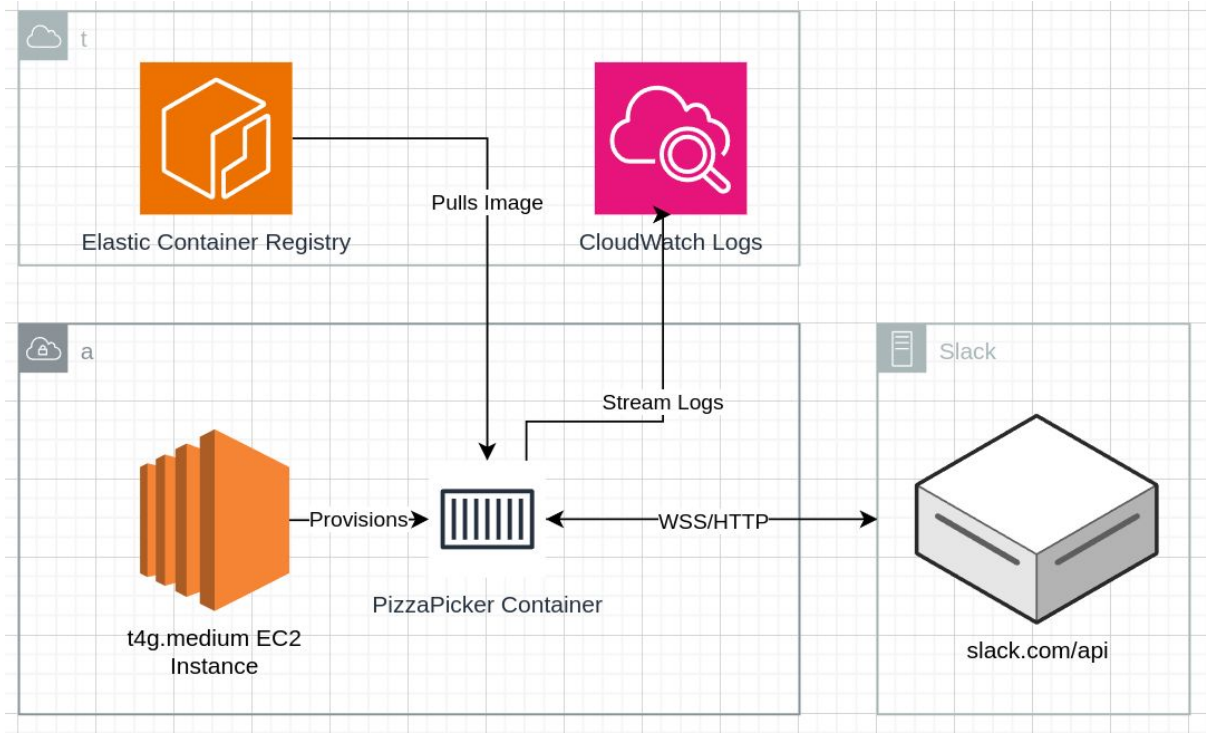
```rust
async fn handle_slash_command(
    message: incoming::Incoming<incoming::SlashCommandIncomingMessage>,
) -> outgoing::SlackOutgoingMessage {
    let spin_mode = match message.payload.command.as_str() {
        "/spin" => RouletteFilter::All,
        "/spin-vegan" => RouletteFilter::Vegan,
        "/spin-vegetarian" => RouletteFilter::Vegetarian,
        _ => {
            tracing::warn!("Received unknown command: {}", message.payload.command);
            return outgoing::SlackOutgoingMessage::Empty(outgoing::Outgoing::new(
                message.envelope_id,
                None,
            ));
        }
    };

    let pizza = get_random_pizza(spin_mode);
    let mention = mention_user(message.payload.user_id);
    let fortune_phrase = get_random_fortune_phrase();
```

SHIP IT

AWS + Terraform

- Docker Container inni EC2 på AWS
- Deler plass med flere andre tjenester



Takk for meg :D

Tuesday, 19 March

Brage A 14:47
/spin

PizzaPicker APP 14:47
Gratulerer, du har fått V-GF18 VEGETARIANEREN VEGANSK

V-GF18 VEGETARIANEREN VEGANSK er en pizza med Nyhet! Ny oppskrift: Vegansk osteerstating, tomatsaus, sjampinjong, rød paprika, gul paprika og vårløk. På 30 cm melke- og glutenfri bunn. (Allergener: selleri)

💀 1

2 replies Last reply 8 days ago

Brage A 14:47
ah hell naw

/spin

PizzaPicker APP 14:47
Gratulerer, du har fått 10 PEPPERSVENNEN

10 PEPPERSVENNEN er en pizza med Ost, tomatsaus, pepperbiff i strimler, sjampinjong, løk og rød paprika. (Allergener: hvete, melk, sennep, selleri)

Brage A 14:48
NEVER BACK DOWN NEVER WHAT

/spin

PizzaPicker APP 14:48
Gratulerer, du har fått 11 FLAMMEN

11 FLAMMEN er en pizza med Ost, tacosaus, kjøttdeig, nachoschips og jalapeños. (Allergener: hvete, melk)

Brage A 14:48
LETS GOOOOOOO