

Tripletex
API Meetup

Tips and tricks

Tripletex API Meetup v3





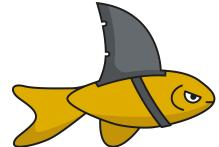
BOND



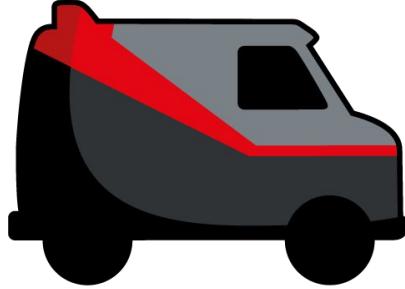
**PIGGY BANK
CREW**



**ANGRY
NERDS**



GOLDSHARK



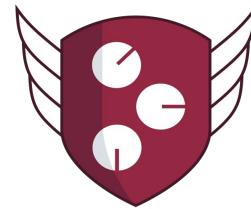
A-TEAM



BEEHIVE



AGRO



TSK



**KLAN ROCK
& REGNSKAP**

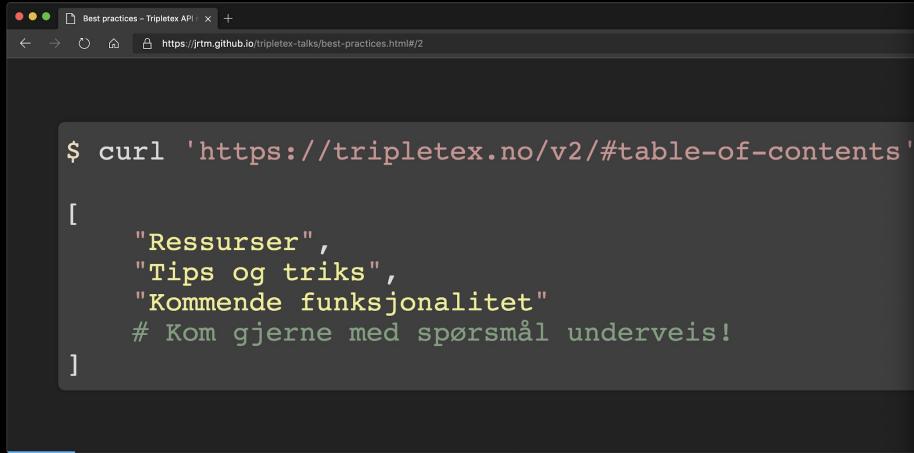


Julian Ravn Thrap-Meyer
Tech Lead API

Previously



Meetup v1



```
$ curl 'https://tripletex.no/v2/#table-of-contents'  
[  
  "Ressurser",  
  "Tips og triks",  
  "Kommende funksjonalitet"  
  # Kom gjerne med spørsmål underveis!  
]
```

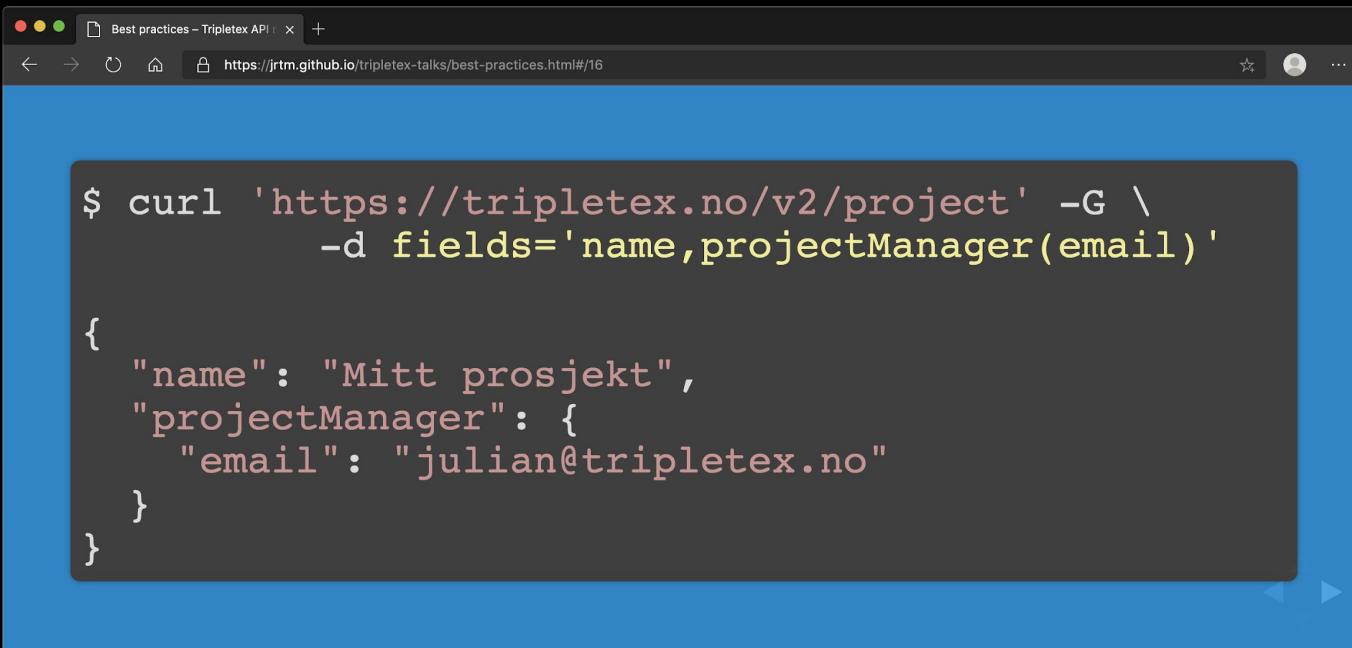
github.com/Tripletex



Meetup v2

Tripletex
API Meetup

Fields



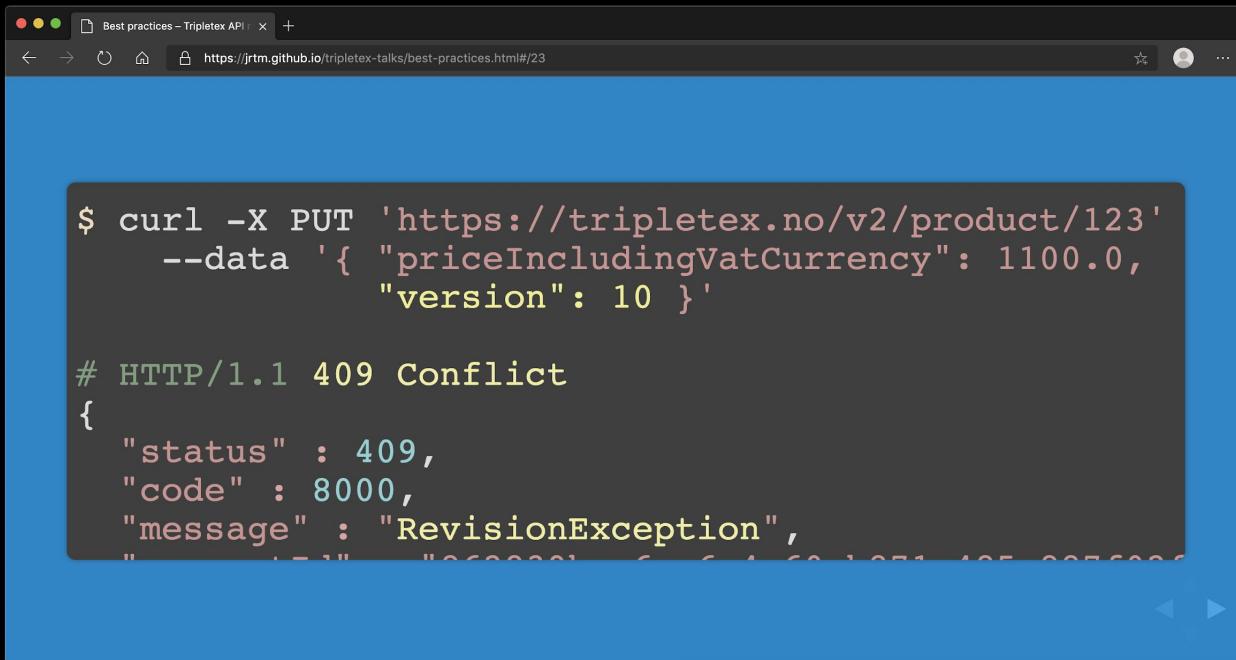
A screenshot of a web browser window titled "Best practices – Tripletex API". The URL is <https://jrtm.github.io/tripletex-talks/best-practices.html#/16>. The page content displays a curl command and its corresponding JSON response.

```
$ curl 'https://tripletex.no/v2/project' -G \
-d fields='name,projectManager(email)'

{
  "name": "Mitt prosjekt",
  "projectManager": {
    "email": "julian@tripletex.no"
  }
}
```



Version



A screenshot of a web browser window titled "Best practices – Tripletex API". The URL in the address bar is <https://jrtm.github.io/tripletex-talks/best-practices.html#/23>. The page content displays a command-line example and its JSON response.

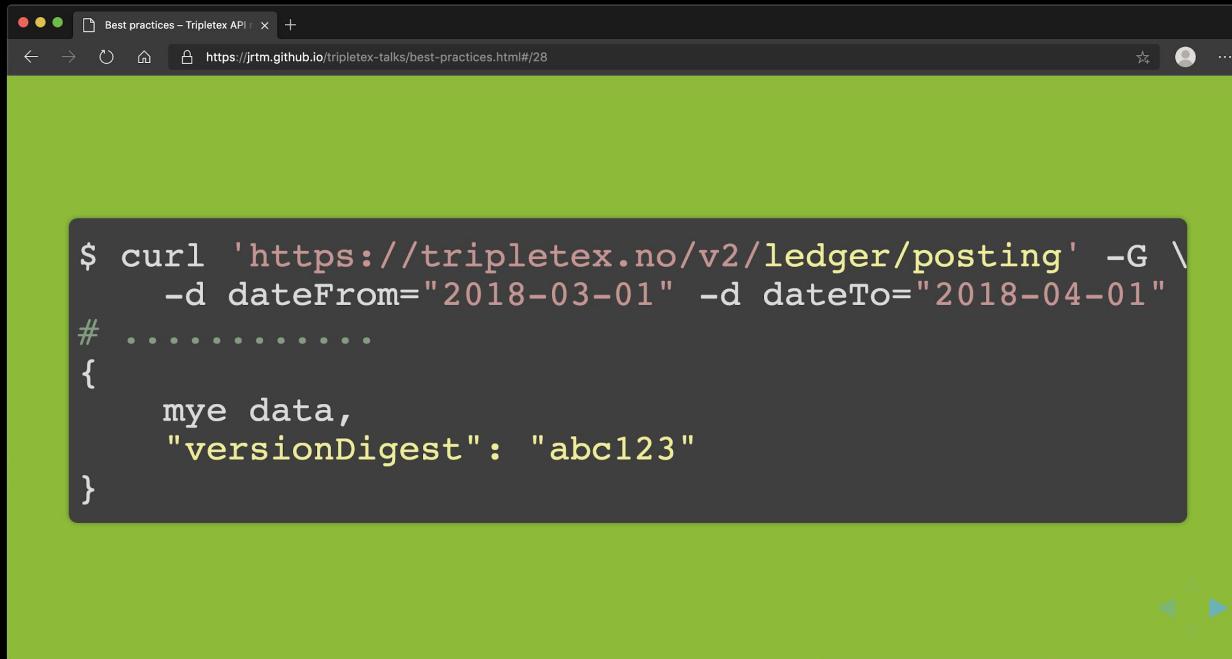
```
$ curl -X PUT 'https://tripletex.no/v2/product/123'
--data '{ "priceIncludingVatCurrency": 1100.0,
          "version": 10 }'

# HTTP/1.1 409 Conflict
{
  "status" : 409,
  "code" : 8000,
  "message" : "RevisionException",
  "error" : "The product version 10 has been updated since you last read it. Please try again."}
```



Tripletex
API Meetup

Checksum



A screenshot of a web browser window titled "Best practices – Tripletex API". The URL in the address bar is [https://jrtm.github.io/tripletex-talks/best-practices.html#/28](https://jrtm.github.io/tripletex-talks/best-practices.html#/). The page content is mostly blank, but a dark gray box contains a curl command:

```
$ curl 'https://tripletex.no/v2/ledger/posting' -G \
      -d dateFrom="2018-03-01" -d dateTo="2018-04-01"
#
{
  mye data,
  "versionDigest": "abc123"
}
```



Checksum

Best practices – Tripletex API

LITT SENERE ...

```
$ curl 'https://tripletex.no/v2/ledger/posting' -G \
-d dateFrom="2018-03-01" -d dateTo="2018-04-01"
-H "If-None-Match: abc123"

# HTTP/1.1 304 Not Modified
{
    # veldig kjapt, uten data
}
```



api.tripletex.io

The image displays two side-by-side browser windows. The left window shows the 'Logg inn - Tripletex' page at <https://api.tripletex.io/execute/login>. It features a logo, input fields for 'Brukernavn' and 'Passord', a blue 'LOGG INN' button, a 'GLEMT PASSORD?' link, and a 'INTEGRASJONSMILJØ' section with links for 'Bestill her eller prøv gratis'. A green arrow points from the bottom of this window towards the right window. The right window shows the 'Tripletex integrasjonsmiljø' page at <https://api.tripletex.io/execute/integrationEnvironment?site=no>. It has a navigation bar with four items (1, 2, 3, 4), a 'Velkommen til Tripletex' header, a welcome message about using the integration environment, and a callout bubble stating 'Du kan finne dokumentasjon og annen hjelpe på vår GitHub-side.' A blue 'KOM I GANG' button is at the bottom. The background of the right window features abstract puzzle piece graphics.

Webhooks

The screenshot shows a GitHub repository page for `tripletex-api2/docs/webhooks`. The page includes a table of contents, a list of available events, and examples of webhook payloads.

Event Type	Description
Invoice charged	Invoice charged
Order created	Order created
Order updated	Order updated
Order deleted	Order deleted
Product created	Product created
Product updated	Product updated
Product deleted	Product deleted
Voucher created	Voucher created
Voucher deleted	Voucher deleted
Voucher updated	Voucher updated (also triggered when changing the voucher's posting rule)
Customer created	Customer created
Customer updated	Customer updated
Customer deleted	Customer deleted
Supplier created	Supplier created
Supplier updated	Supplier updated
Supplier deleted	Supplier deleted
Project created	Project created
Project updated	Project updated
Project deleted	Project deleted
Customer created	Customer created
Customer updated	Customer updated
Customer deleted	Customer deleted
Supplier created	Supplier created
Supplier updated	Supplier updated
Supplier deleted	Supplier deleted
Order creation confirmation event	Let us know if you have any suggestions!

Events:

- Create
- Update
- Delete
- Custom events

Subscribing to an event

```
POST https://tripletex.no/v2/event/subscription
{
  "event": "product.create",
  "target": "https://your.receiver",
}
```

- ## Events:
- Create
 - Update
 - Delete
 - Custom events



Resources



github.com/tripletex

The screenshot shows a GitHub README page for the 'Tripletex API 2' repository. The page title is 'Tripletex API 2'. It contains three main sections: 'Resources', 'Documentation', and 'Code examples'. The 'Resources' section lists links to Swagger API documentation, a Test environment (with a note about self-service creation of integration test accounts), a Changelog, Frequently asked questions, About API 2.0, and a Workplace API group (open for everyone, with a link to request an invitation). The 'Documentation' section lists links to Webhooks and Zapier app (beta). The 'Code examples' section lists a Bash (shell script) example, which includes a link to 'Create authentication user...'. The URL in the browser bar is https://github.com/Tripletex/tripletex-api2.

GitHub - Tripletex/tripletex-api2

https://github.com/Tripletex/tripletex-api2

README.md

Tripletex API 2

Resources

- [Swagger API documentation](#) (Always up-to-date!)
- [Test environment](#) (Self-service creation of integration test accounts)
 - [Register new test integration](#)
- [Changelog](#)
- [Frequently asked questions](#)
- [About API 2.0](#)
- [Workplace API group](#) (open for everyone, [request an invitation here](#))

Documentation

- [Webhooks](#)
- [Zapier app \(beta\)](#)

Code examples

- [Bash \(shell script\)](#)
 - [Create authentication user...](#)



Swagger

Tripletex API 2.0

project/period

- project/task >
- purchaseOrder >
- reminder >
- salary/compilation >
- salary/payslip >▼

GET /salary/payslip [BETA] Find payslips corresponding with sent data.

Try it out

Parameters

Name	Description
id string (query)	List of IDs id - List of IDs
employeeId string (query)	List of IDs employeeId - List of IDs

Tripletex API 2.0

https://tripletex.no/v2-docs/#salary%2Fpayslip/s...

ListResponsePayslip ▼ {
 fullResultsSize integer(\$int32)

 [DEPRECATED] Indicates whether there are more values available. Note: The value is not exact

 from integer(\$int32)
 count integer(\$int32)
 versionDigest string

 Used to know if the paginated list has changed.

 values ▼ {
 [Payslip ▼ {
 id integer(\$int32)
 version integer(\$int32)
 changes [...]
 url string
 readOnly: true
 transaction SalaryTransaction > {...}
 employee* Employee > {...}
 date string

 Voucher date.
 year integer(\$int32)
 month integer(\$int32)
 specifications > [...]
 vacationAllowanceAmount number
 readOnly: true
 grossAmount number
 amount number
 number integer(\$int32)
 readOnly: true
 minimum: 0
 }]
 }
}



Workplace

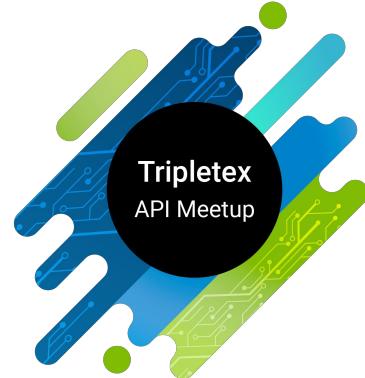
The screenshot shows the Tripletex API group page on Facebook Workplace. The header features the Tripletex logo and a decorative background of two stylized blue pencils. Below the header, the group name "Tripletex API" is displayed, along with a red "Flerbedrift" button and a "Lukket gruppe" link. The navigation bar includes tabs for "Om", "Innlegg" (highlighted in blue), "Filer", and "Mer". A search bar at the top right says "Søk i gruppe ...". The main content area contains a message input field with placeholder "Skriv noe ...", several interaction buttons ("Legg til fil", "Legg til bilde...", "Tagg kollegaer", "..."), and a note about members from outside the group. At the bottom, a "NY AKTIVITET" section lists an event titled "Apí Integratoresen" from December 23, 2019, with a description of the API integrator session.



teknisk-support



Live demo!





- **# List available webhook events**
- **GET /event**
- **# Get example payloads**
- **GET /event/{type}**
- **# Create webhook subscription**
- **POST /event/subscription**
- **# Cancel webhook subscription**
- **DELETE /event/subscription/{id}**



Takk! :-)

Spørsmål?



PAUSE

Integration showcase

Tripletex API Meetup v3





RecMan - Tripletex API

Victor Franzén
RecMan AS



Om RecMan

Etablert 2013

Cloudbasert ERP - System

Bemannning

Rekruttering

CRM

Fakturering og lønn

Konnektivitet & API



Integrasjon mellom RecMan-Tripletex

Historie fra 2014 CSV -> API

Felles problemløsning

Hva integrerer vi mot Tripletex

Erfarenheter

Samarbeid for at vinne nye kunder



Demo RecMan - Tripletex API

VIDEO



Takk!

Spørsmål?

Victor Franzén
RecMan AS



PAUSE



Goliath Systemer





Et selskap som utvikler digitale arbeidsverktøy som forenkler arbeidsflyt og datafangst på tvers av systemer.

GO DIGITAL

Digital styringsverktøy

GO ADVISOR

Rådgivning og
bistand

GO DRIVE

Elektronisk kjørebok



Samarbeid

For snart 10 måneder siden bestemte vi oss, at Goliath må integreres med et fullverdig regnskapssystem.

Og valget falt naturlig på **Tripletex** etter litt research.

Fantastiske mennesker og eksepsjonell oppfølging, kombinert med at Tripletex har et system som dekker de aller fleste områder.





+



Goliath og tripletex er etter 6 mnd arbeid blitt integrert sammen, på et høyt nivå hvor det er mange elementer som snakker sammen **helt automatisk**.

Føles som magi 





Hvordan gjorde vi det?

Når noe skal integreres med andre systemer er det mange detaljer som må gjennomgåes. Vi begynte med og etablere et godt samarbeid og granske API dokumentasjonen som finnes på nett.

Som produktansvarlig var det naturlig å mestre Tripletex som et system, slik at ved onboarding av nye kunder har man forståelse for alle områder samt man kan bistå under hele prosessen. Goliath bruker også Tripletex daglig i egen virksomhet.

Etter mye kartlegging og flere samtaler med Tripletex besluttet vi å lage en omfattende og helhetlig integrasjon som synkroniserer all data som er nødvendig for en smidig arbeidsflyt.





Før synkronisering

Før første synkronisering av eksisterende data kan utføres, må vi klargjøre begge systemer. Når synkronisering startes og hvis noe er galt, får man en oversikt over hva som må utbedres.

Når alt er på plass kan vi starte synkroniseringen, som man kan se grafisk og følge med helt til alt er 100% overført.

Etter at synkronisering er fullført er systemer koblet sammen og all data vil nå automatisk oppdateres umiddelbart begge steder uavhengig hvor du legger informasjon.





Hva synkroniseres

- Avdelinger
- Ansatte
- Kunder og leverandører
- Prosjekter
- Prosjektkategorier
- Maskintimer som ordrelinjer
- Ordre
- Produkter
- Reiseregninger
- Reiseregning på prosjekt
- Statens KM satser
- Bompenger
- Parkering
- Timer til lønn
- Timer på prosjekt
- Timer til fakturering
- Ferie og permisjon – kommer
- Og en rekke andre detaljer innenfor de forskjellige kategoriene



Et eksempel

La oss si at en ansatt registerer 8 arbeidstimer på et prosjekt, med bruk av gravemaskin i 3 av timene + knyttet til en kjøreturen til arbeidsplassen.

Når dette blir lagret blir det opprettet som følger i Goliath timelisten:

1. Arbeidstimer knyttet til prosjekt og aktivitet
2. Maskintimer knyttet til prosjekt og aktivitet
3. KM på kjøretur knyttet mot aktivitet
4. Bompenger knyttet til aktivitet
5. Parkering knyttet til aktivitet

Når timer blir godkjent av en leder i Goliath skjer følgende i Tripletex:

1. Timer sendes til timeliste knytte til korrekt prosjektaktivitet
2. Maskintimer sendes til prosjekt og legges til som en ordrelinje (som et produkt)
3. Det blir opprettet en reiseregning knyttet mot prosjekt og legges både under reiser og utlegg + på reiser og utlegg inne på prosjekt og inneholder fra og til adresse, formål, antall km, korrekt sats basert på innstillinger, bompenger mot korrekt betalingstype og kostnadskategori samt parkerings kostnader.

Dette eksempelet er hos flere av våre kunder reelt og det sies at man sparer ufattelig med tid da dette eksempelet er mye manuelt arbeid og samle inn samt skrive inn.





Økt lønnsomhet

En av de største fordelene med å etablere digitale endringer i virksomheter er **økt lønnsomhet**. Noe alle virksomheter setter stor pris på.

Men det er viktig at virksomhetens eiere, ledere og ansatte er innstilt og villige til og gjøre de endringen som må til for å få på plass gode rutiner og verktøy.

Hvis man investerer tid er dette noe alle kan få til sammen med riktig partner og resultatet vil være overraskende positivt for de fleste over tid.

Et eksempel er en kunde i Oslo, de har 107 ansatte og 15 måneder etter overgangen til nye løsninger og partner har dem hatt en besparelse på ca 2 årsverk så langt.





Sikkerhet

Sikkerhet er viktig for alle, spesielt nå etter mye snakk om GDPR og personvern.

Goliath systemer tar dette på alvor og samarbeider kun med partnere som kan vise til dokumentasjon som bekrefter at data blir behandlet og ivaretatt på en trygg og sikker måte. Og at data i tillegg aldri forlater Europa.

I tillegg har vi en rekke rutiner internt som sørger for at all informasjon blir ivaretatt og våre systemer er bygget med innebygget personvern.

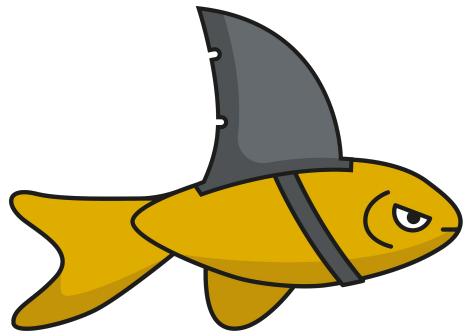


PAUSE

Et blikk fremover

Tripletex API Meetup

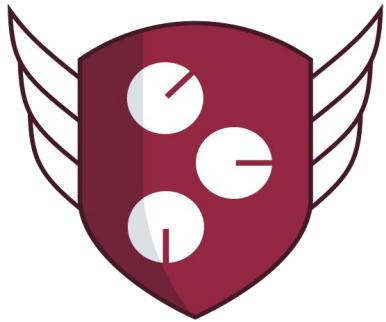




GOLDSHARK

- Varemottak
- Innkjøp
- Forbindelse mellom inngående faktura og innkjøp
- Produktgrupper





TSK



**ANGRY
NERDS**

- Prosjekt, inkludert fakturering til kunde
- Beregning av korrekt sats og antall for kostgodtgjørelse på reiseregning
- Lønnsart på timeføringer

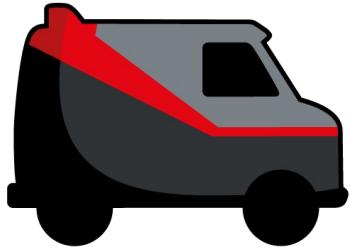




KLAN ROCK & REGNSKAP

- Avtalegiro
- Attestering
- Hente ut gyldige mva-typer
- Vise motpost i hovedbok
- Lukke poster





A-TEAM

- Støtte for OAuth2
- Forenkle uthenting av store mengder data via API
- Forenkle opprettelse av interne integrasjoner





BEEHIVE

- Opprettelse av konto (Tripletex provisjonering)
- Prisberegning (internt)
- Kjøp av moduler (internt)
- App Store for integrasjoner

