

optimising for

FAST FLOW

in Norway's largest bureaucracy

Audun F. Strand



Truls Jørgensen

TL;DR:



NORWEGIAN LABOUR AND WELFARE ADMINISTRATION

assist people into work

child benefits

sickness benefits

unemployment benefits

pensions and more

1/3

of state budget

103 000



**DEV HOURS
IN A SINGLE
RELEASE**

EST. 2006

2016

TL;DR:



NORWEGIAN LABOUR AND WELFARE ADMINISTRATION

assist people into work

child benefits

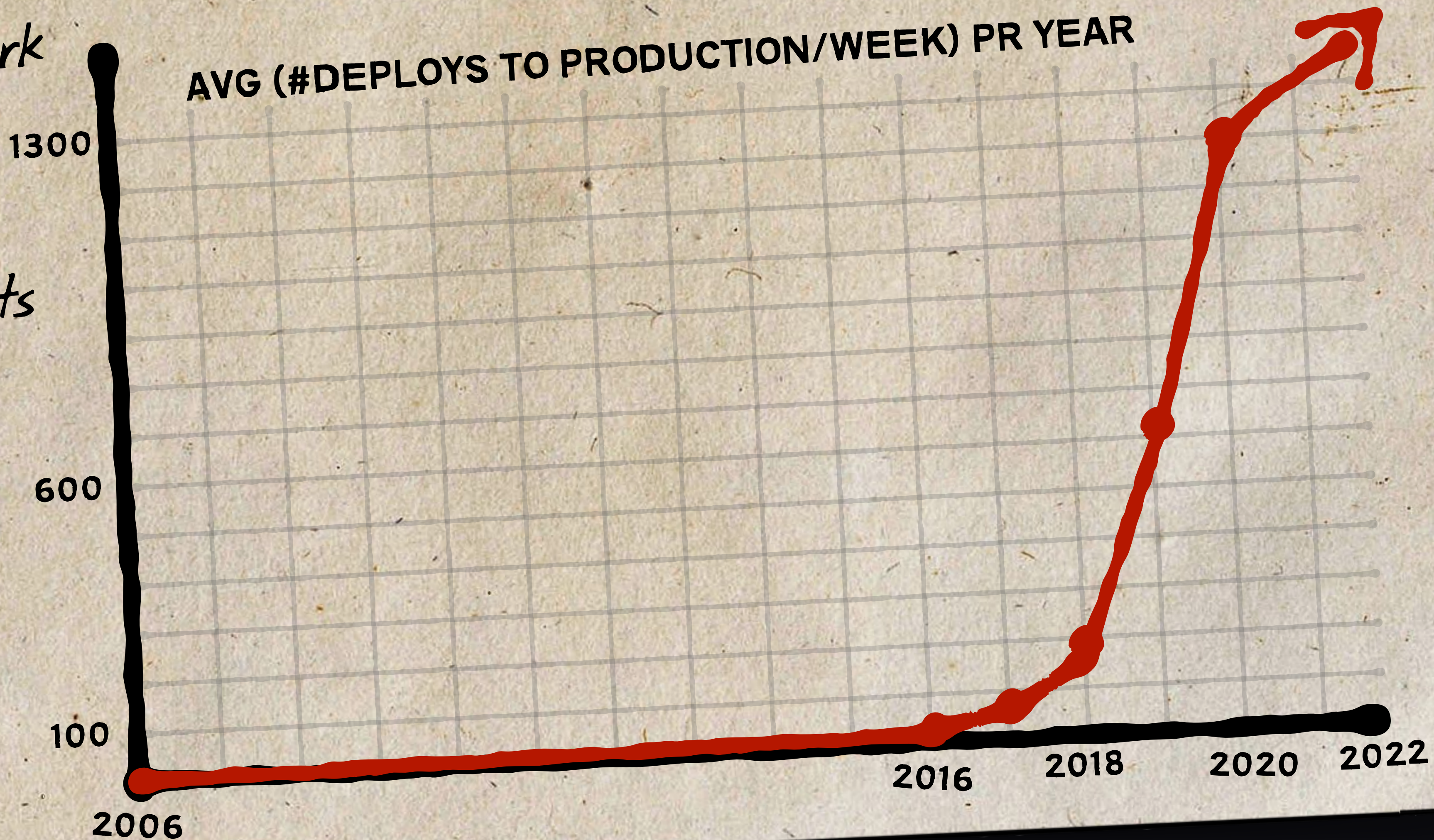
sickness benefits

unemployment benefits

pensions and more

1/3
of state budget

AVG (#DEPLOYS TO PRODUCTION/WEEK) PR YEAR



TL;DR:

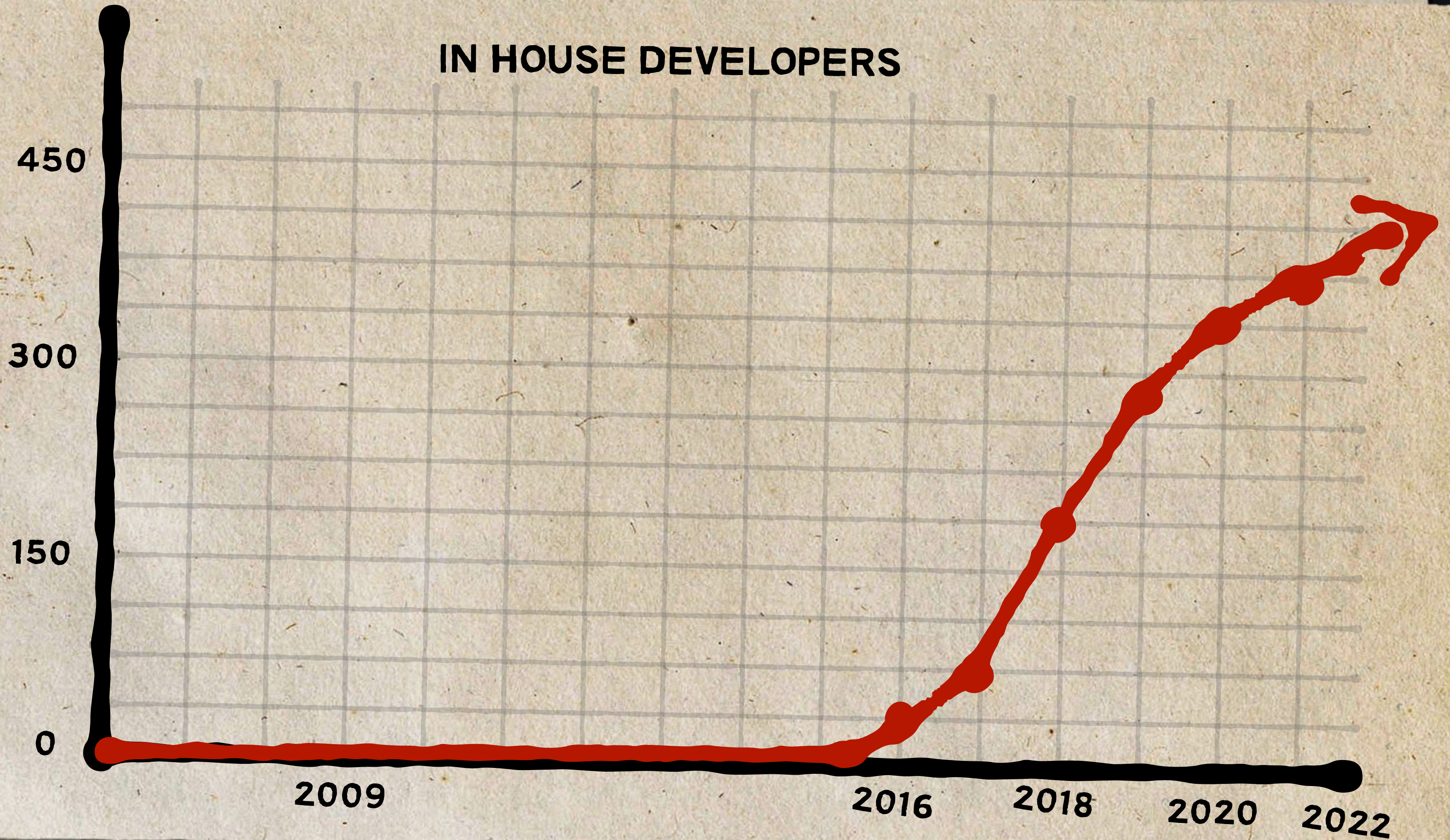


NORWEGIAN LABOUR AND WELFARE ADMINISTRATION

*unemployment benefits
pensions
child benefits..*

1/3
of state budget

IN HOUSE DEVELOPERS



TL;DR:

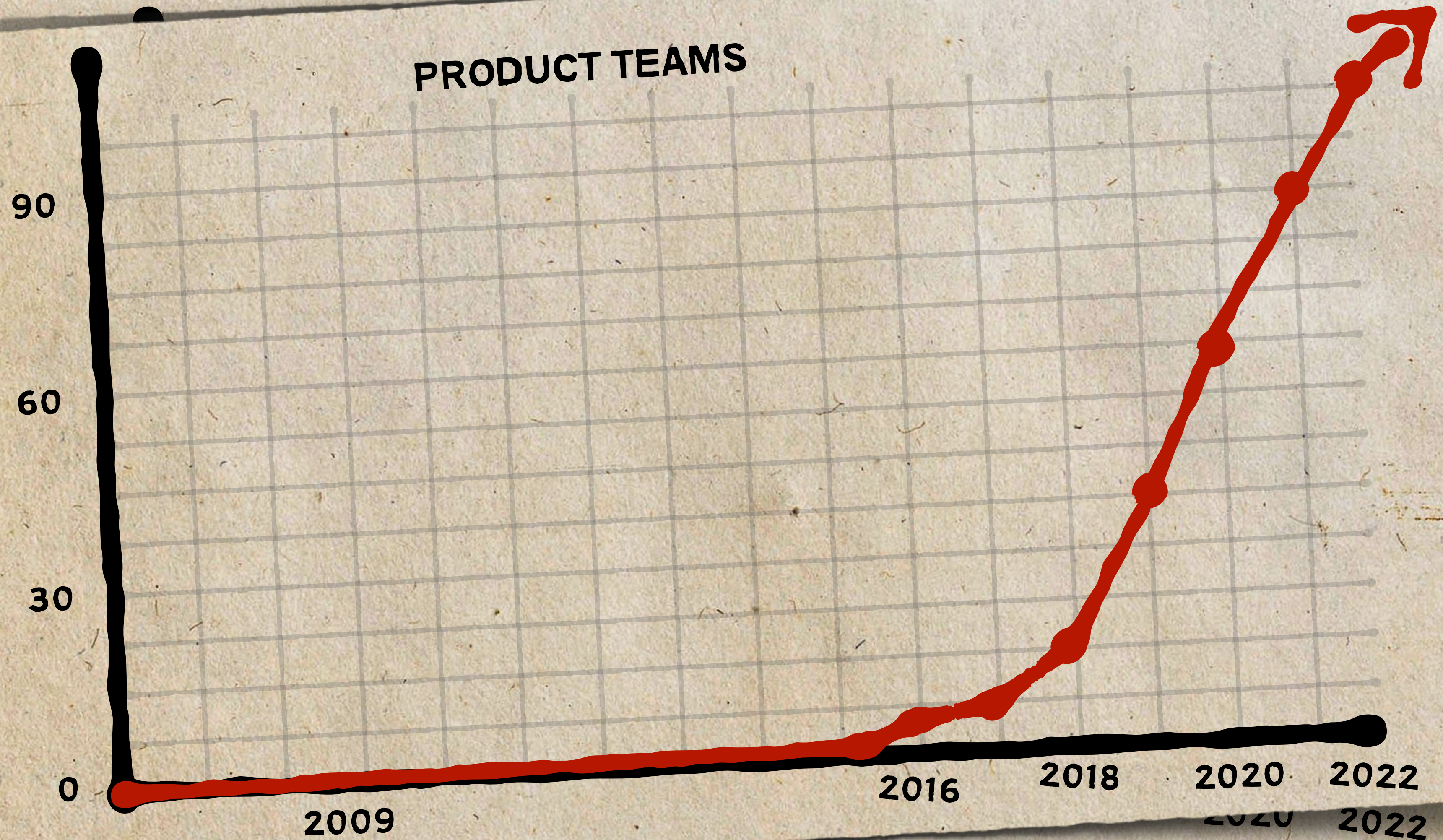
unemployment ben
pensions
child benefits..

1/3
of state budget



NORWEGIAN LABOUR AND WELFARE ADMINISTRATION

PRODUCT TEAMS





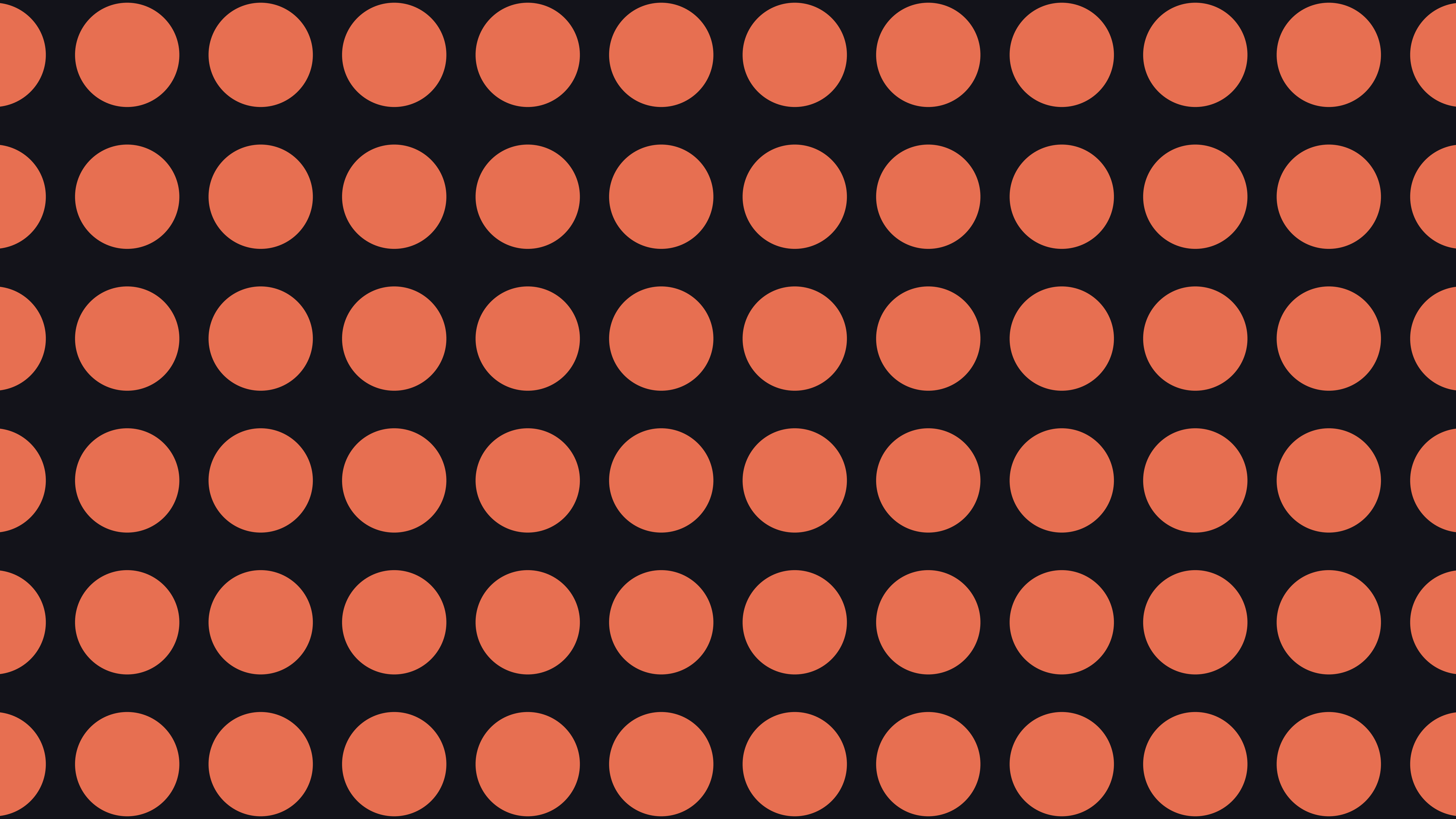
FAST FLOW

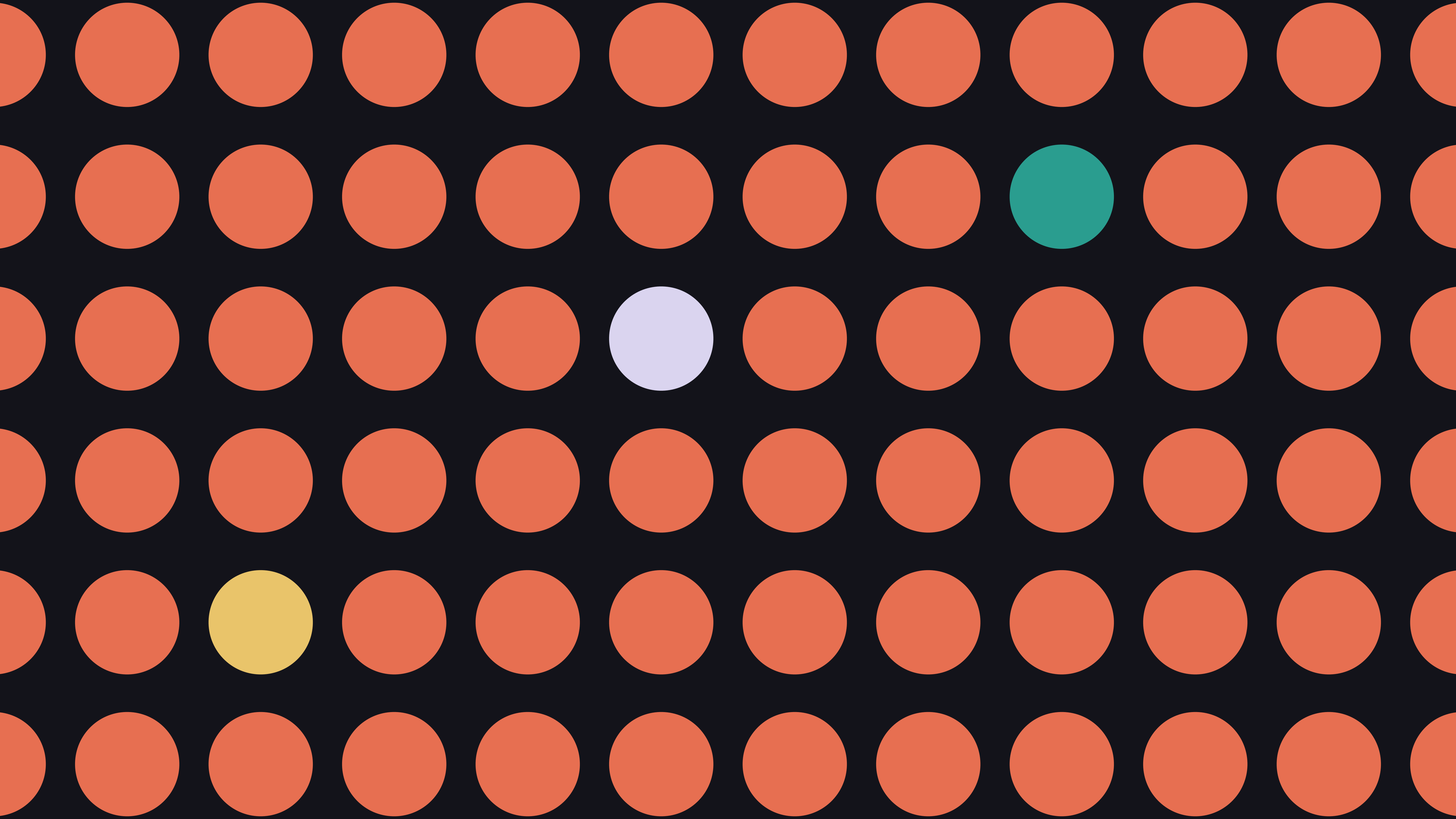
what we left behind

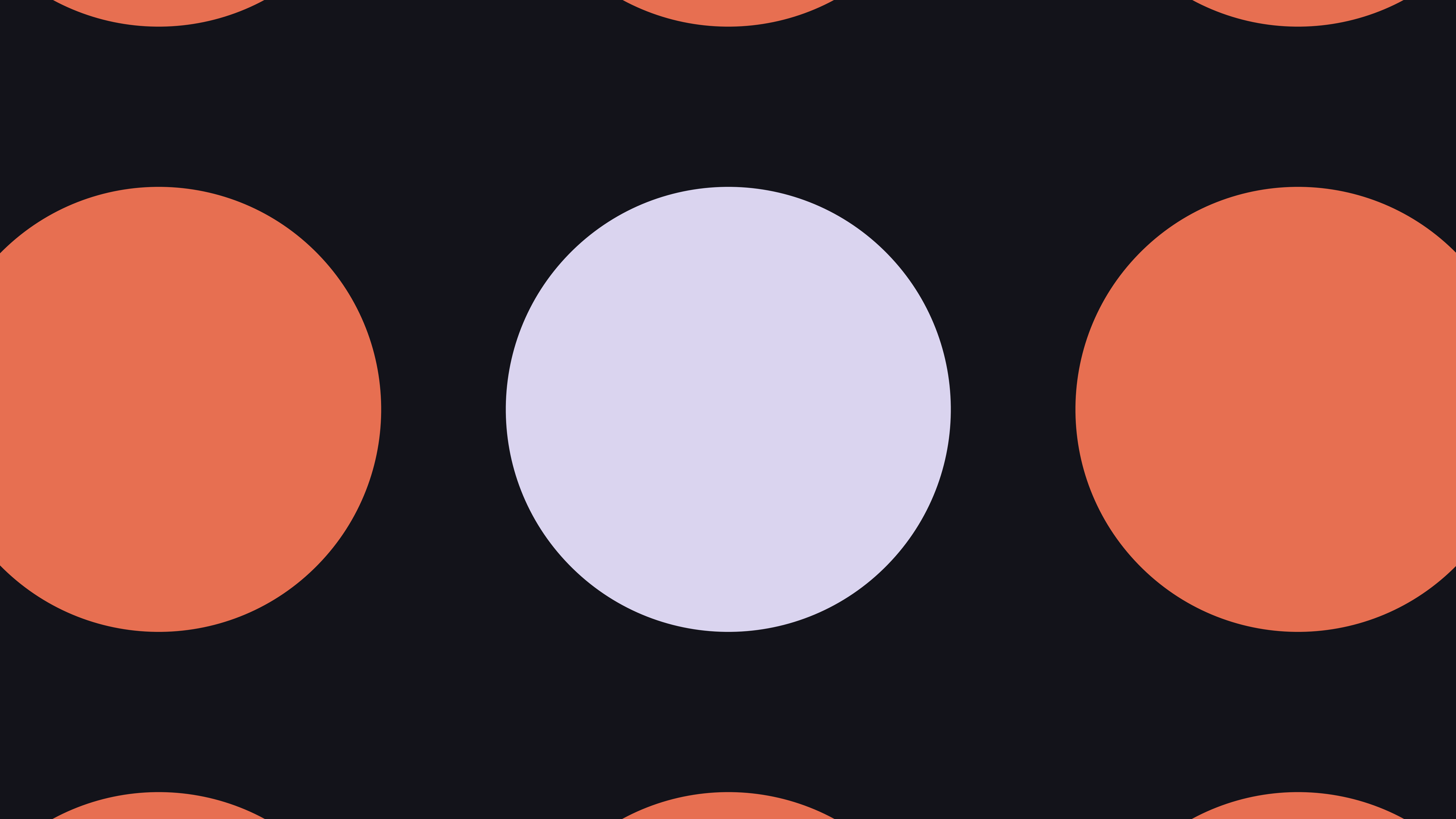


FAST FLOW

what we adopt and prefer







sickness benefit



sickness benefit

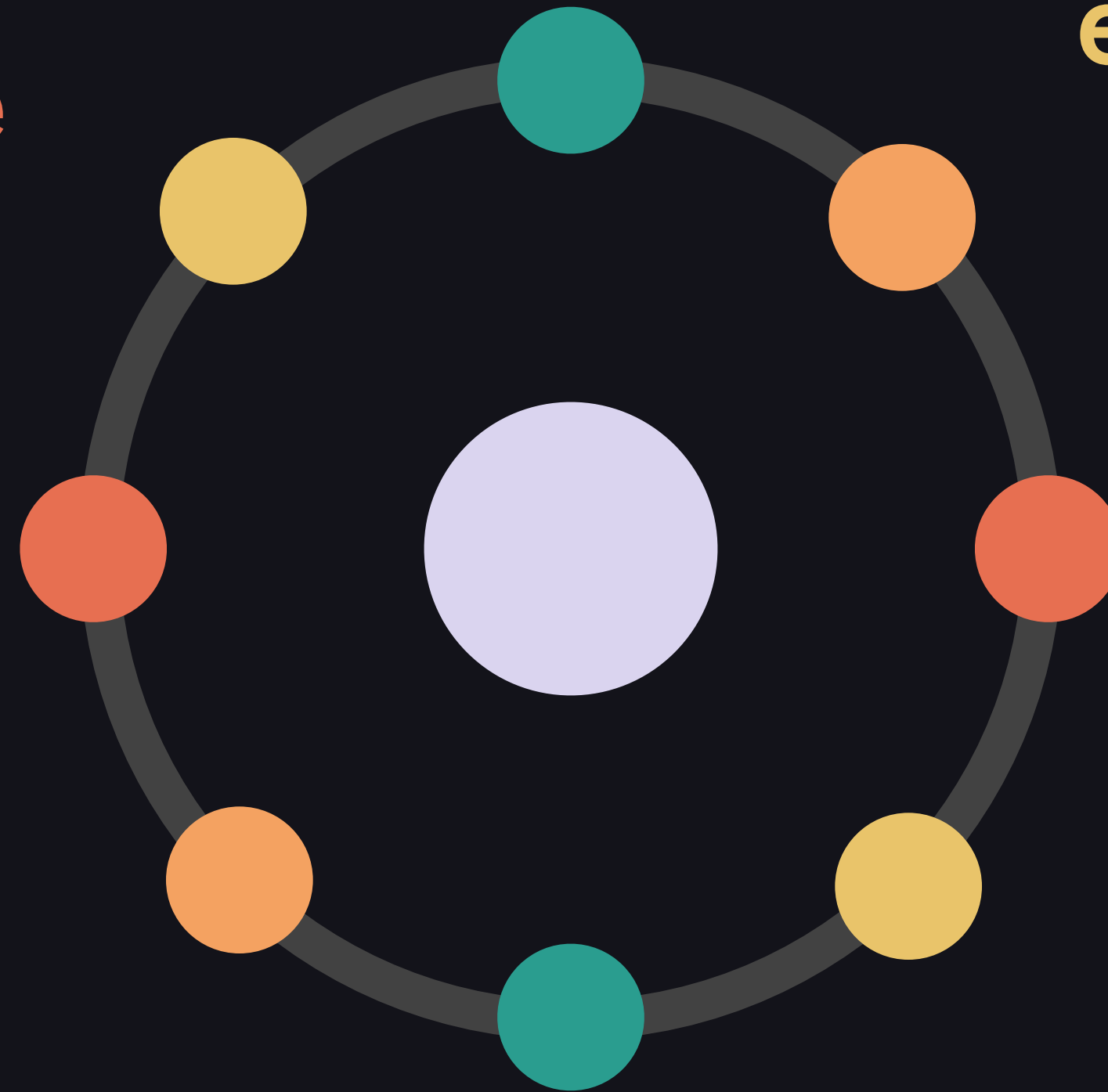


cross functional team

sickness benefit

sustainable system,
adaptive to change

efficient case handling



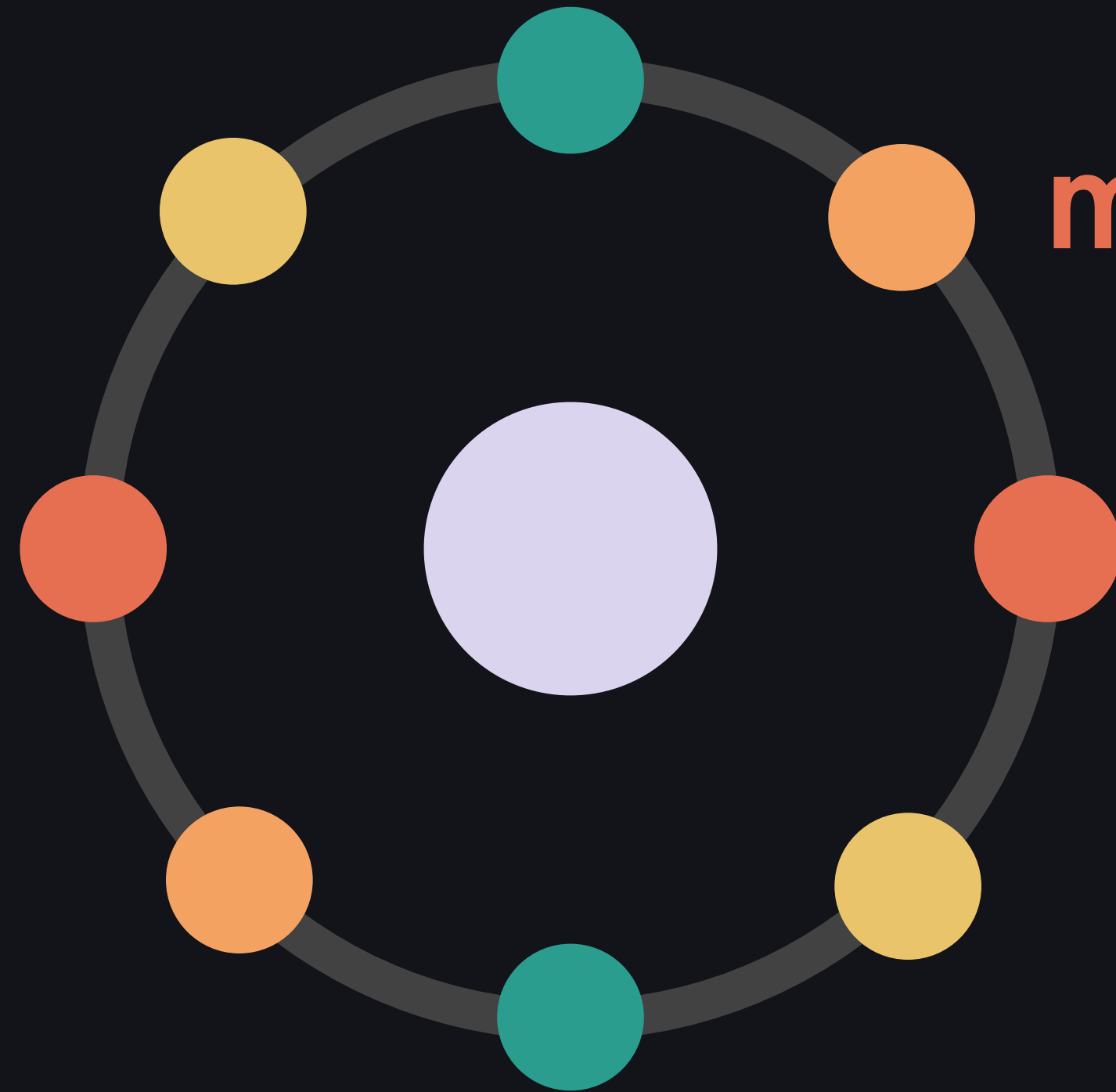
more user friendly service

better compliance

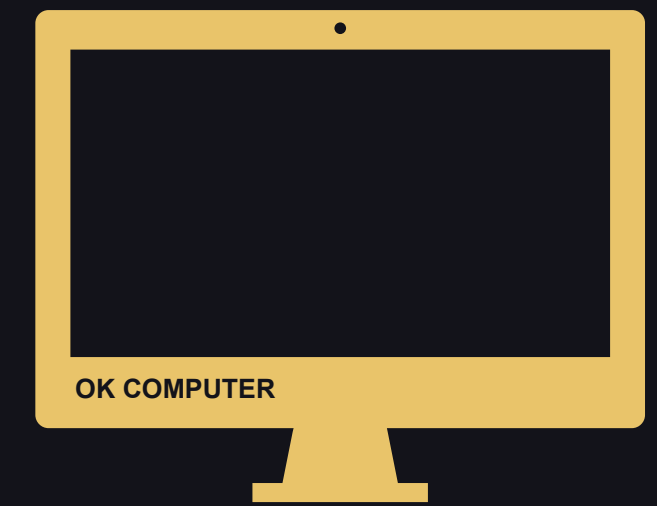
sickness benefit



human
practice



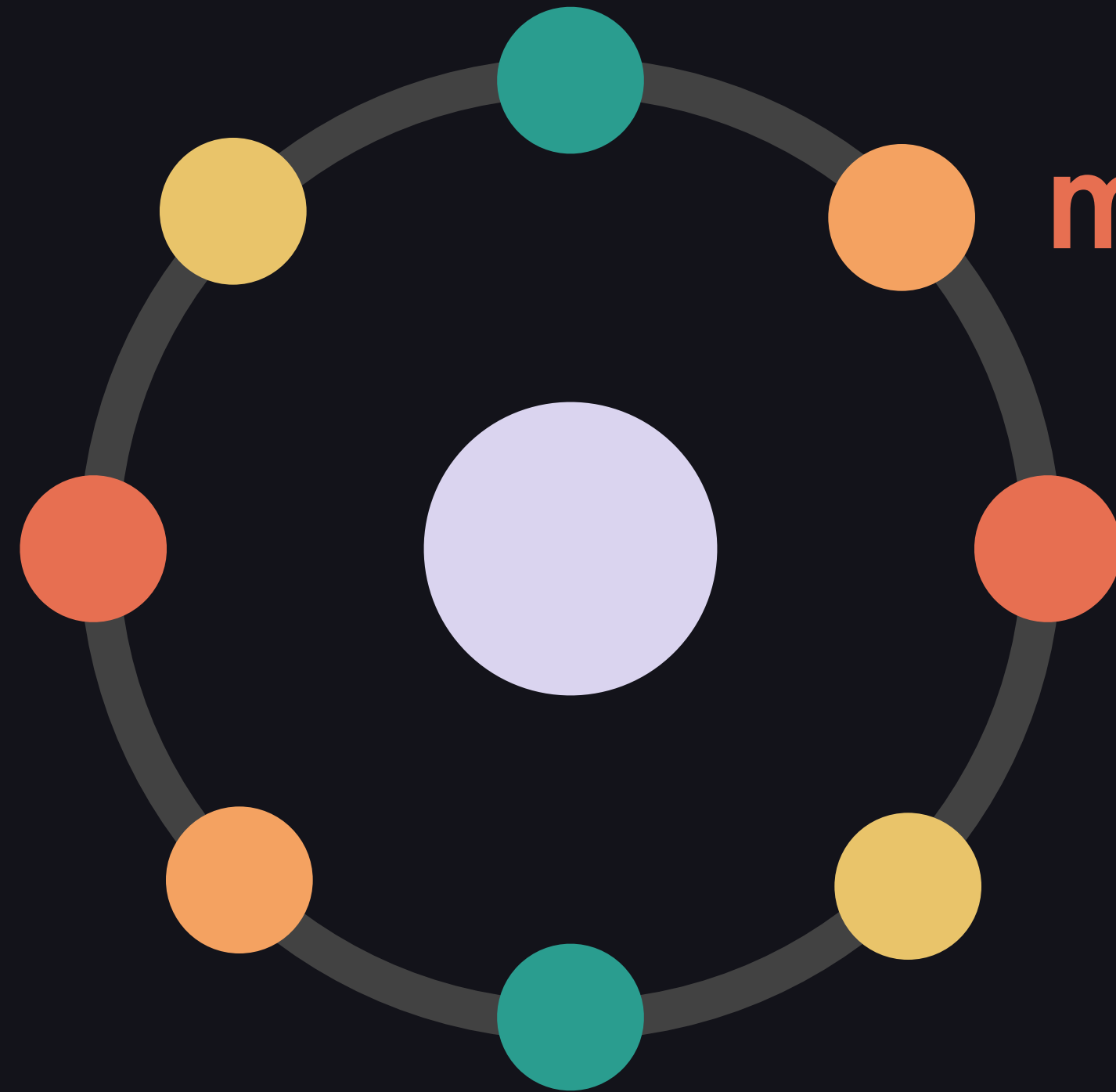
machine
theory



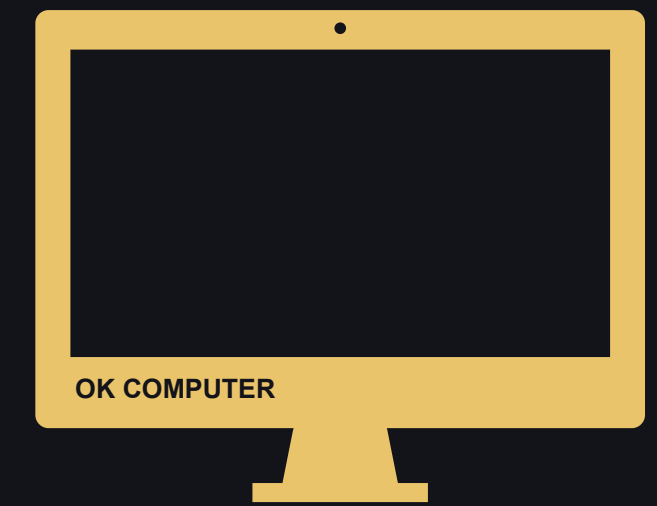
sickness benefit



human
practice



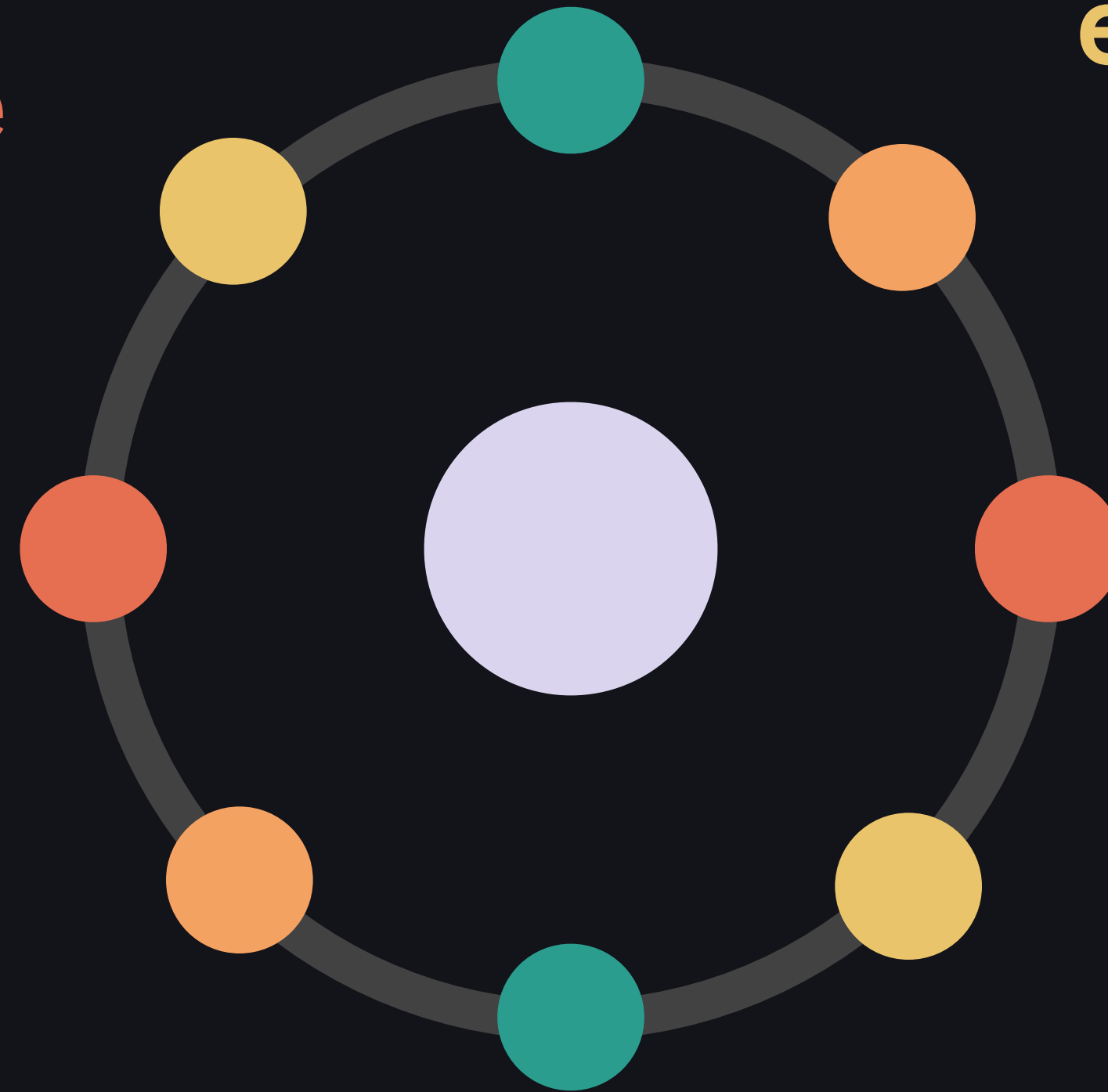
machine
theory



sickness benefit

sustainable system,
adaptive to change

efficient case handling



more user friendly service

better compliance

software development is a

LEARNING PROCESS

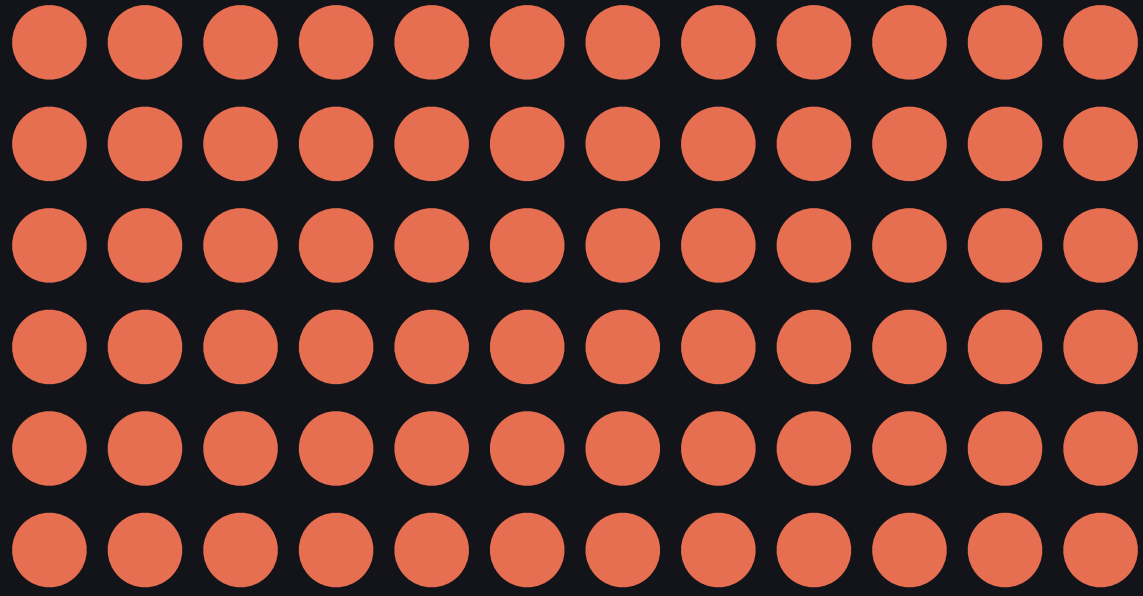
working code is a side effect

-Alberto Brandolini

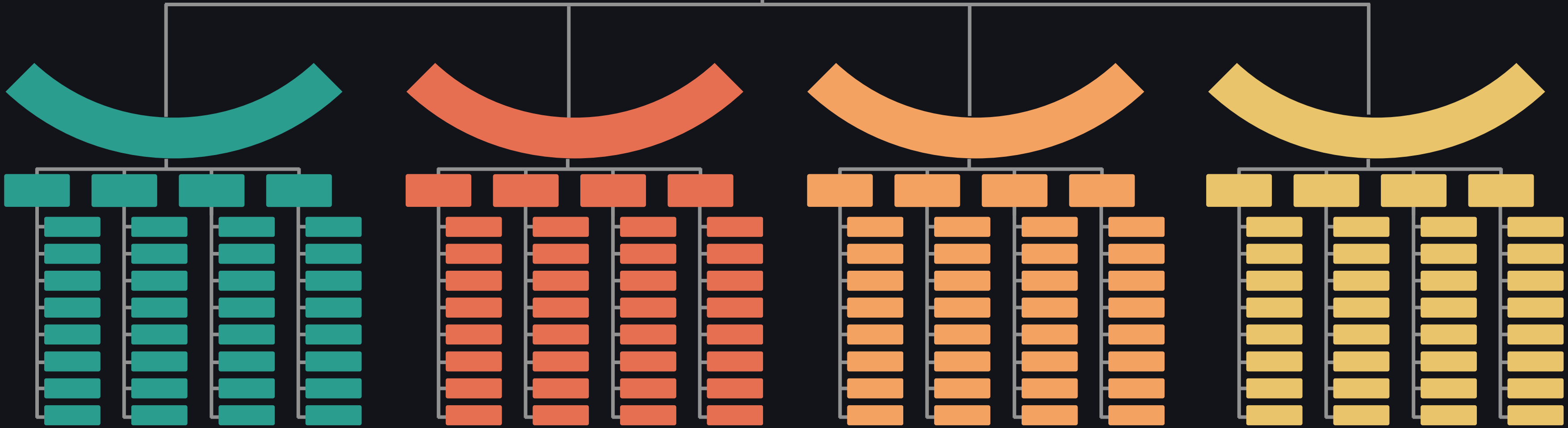
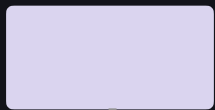


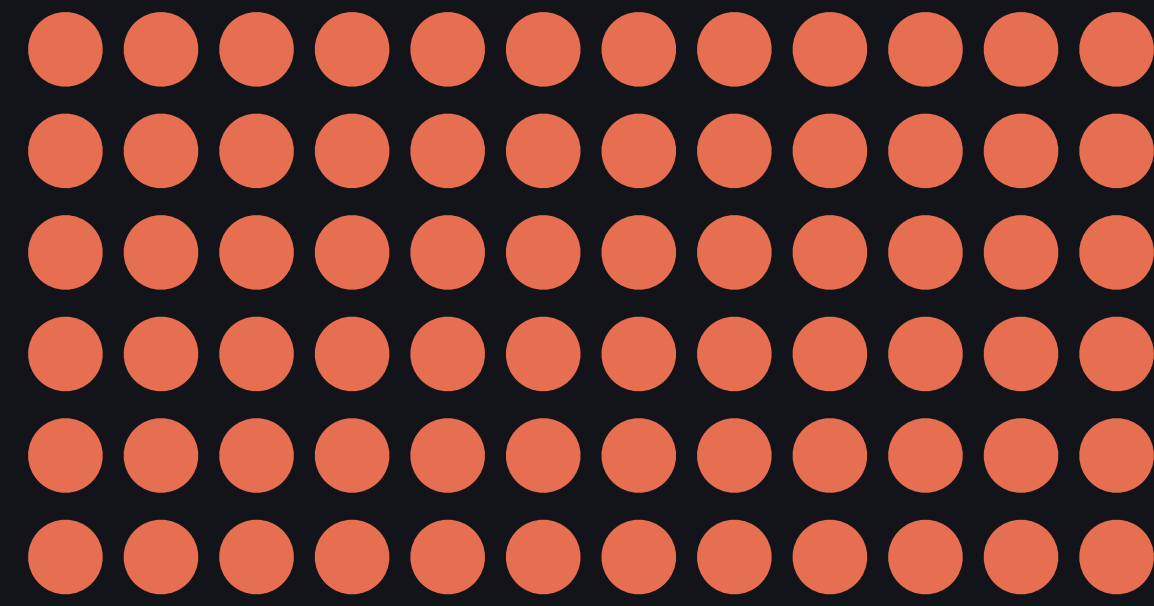
support





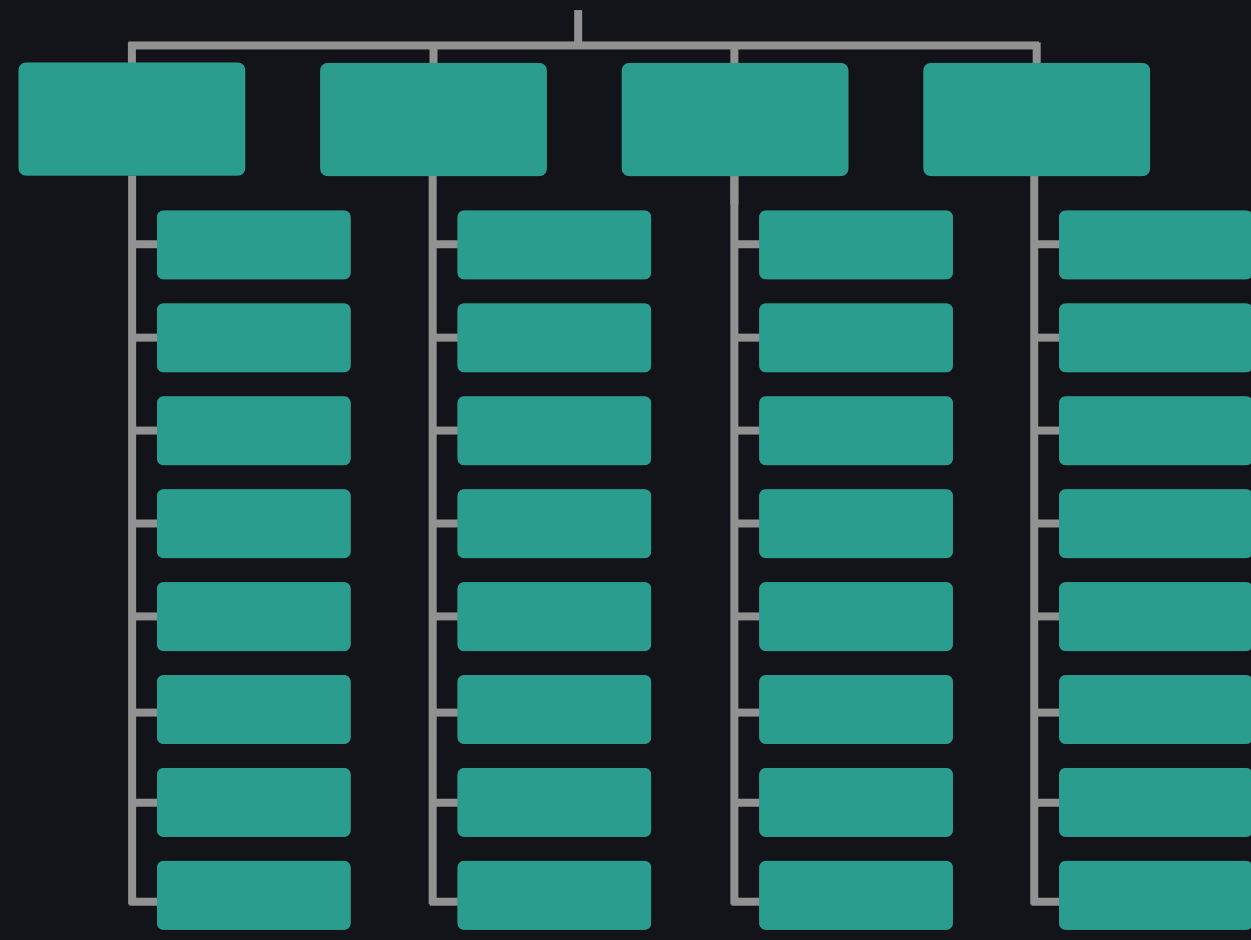
VS



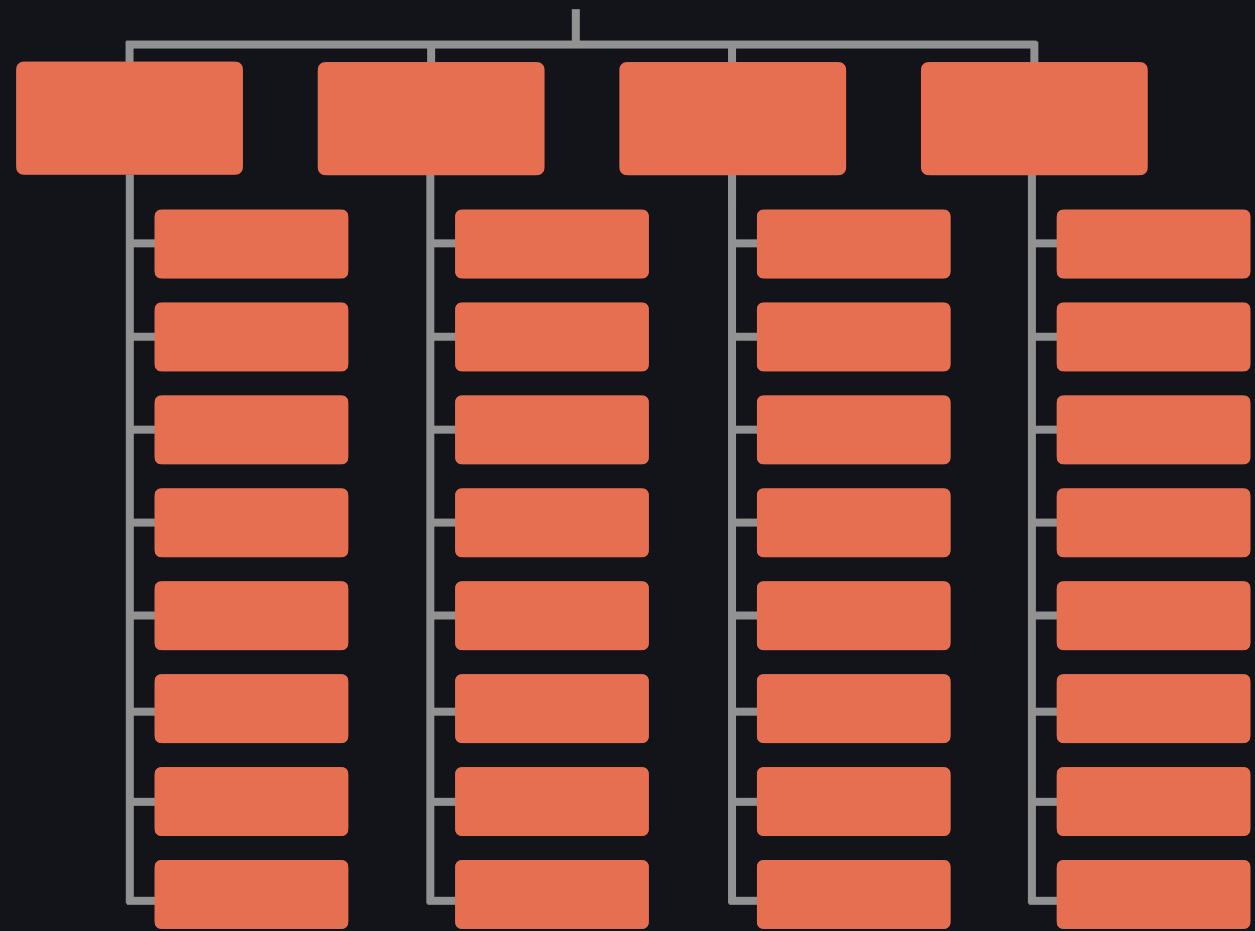
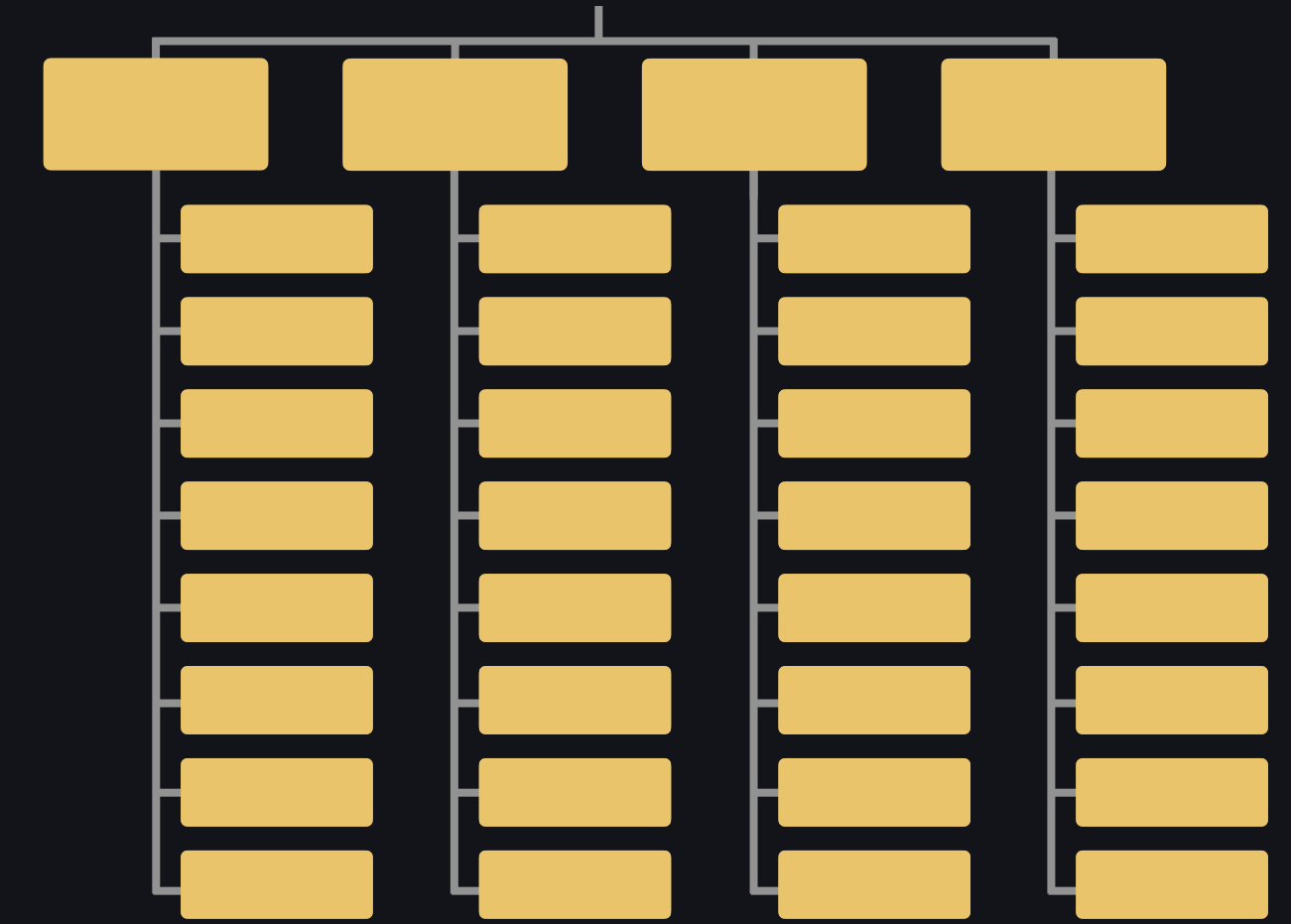


everybody wants to go to heaven
nobody wants to die

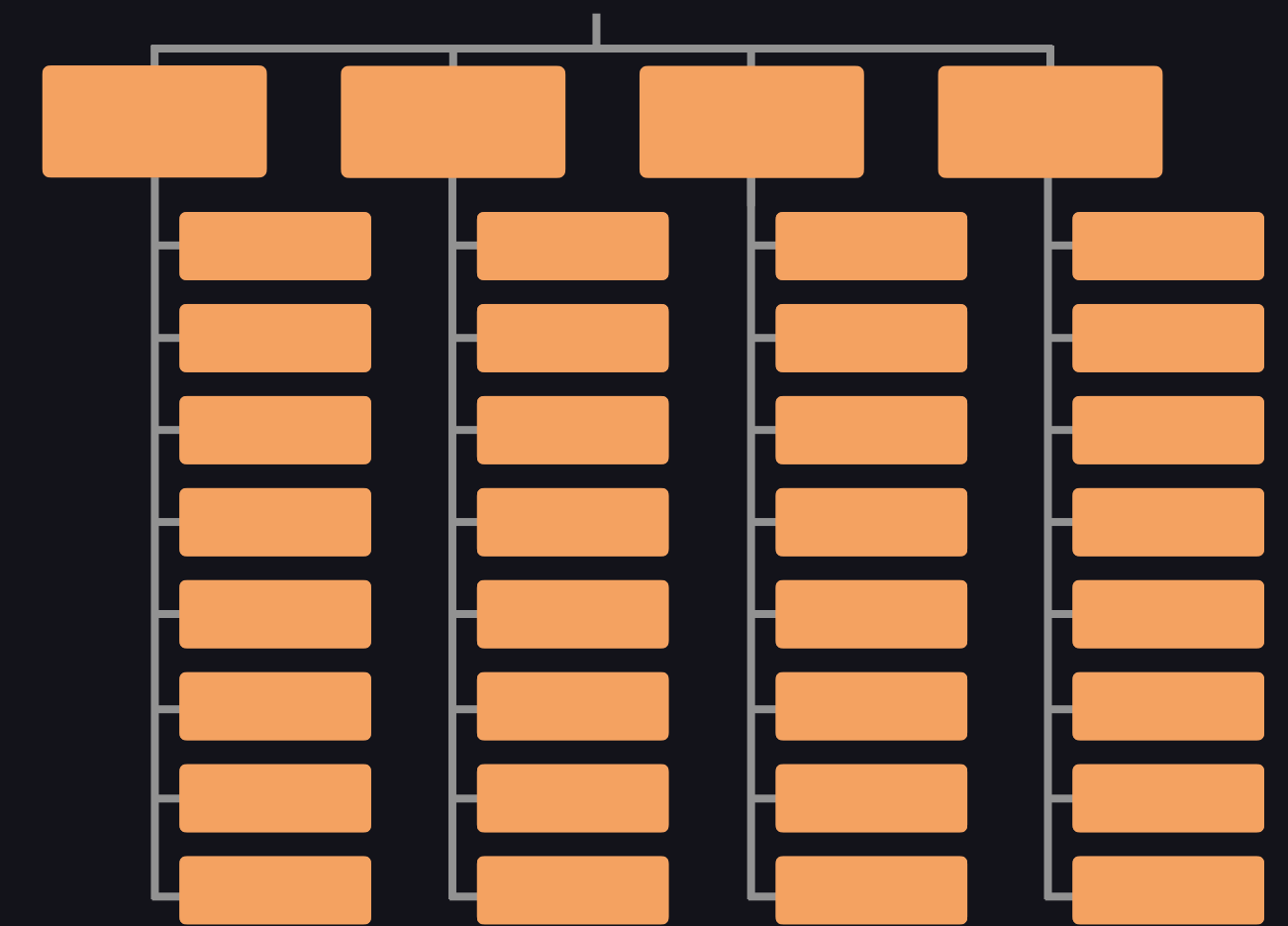




what could these do



to set the teams up for success?

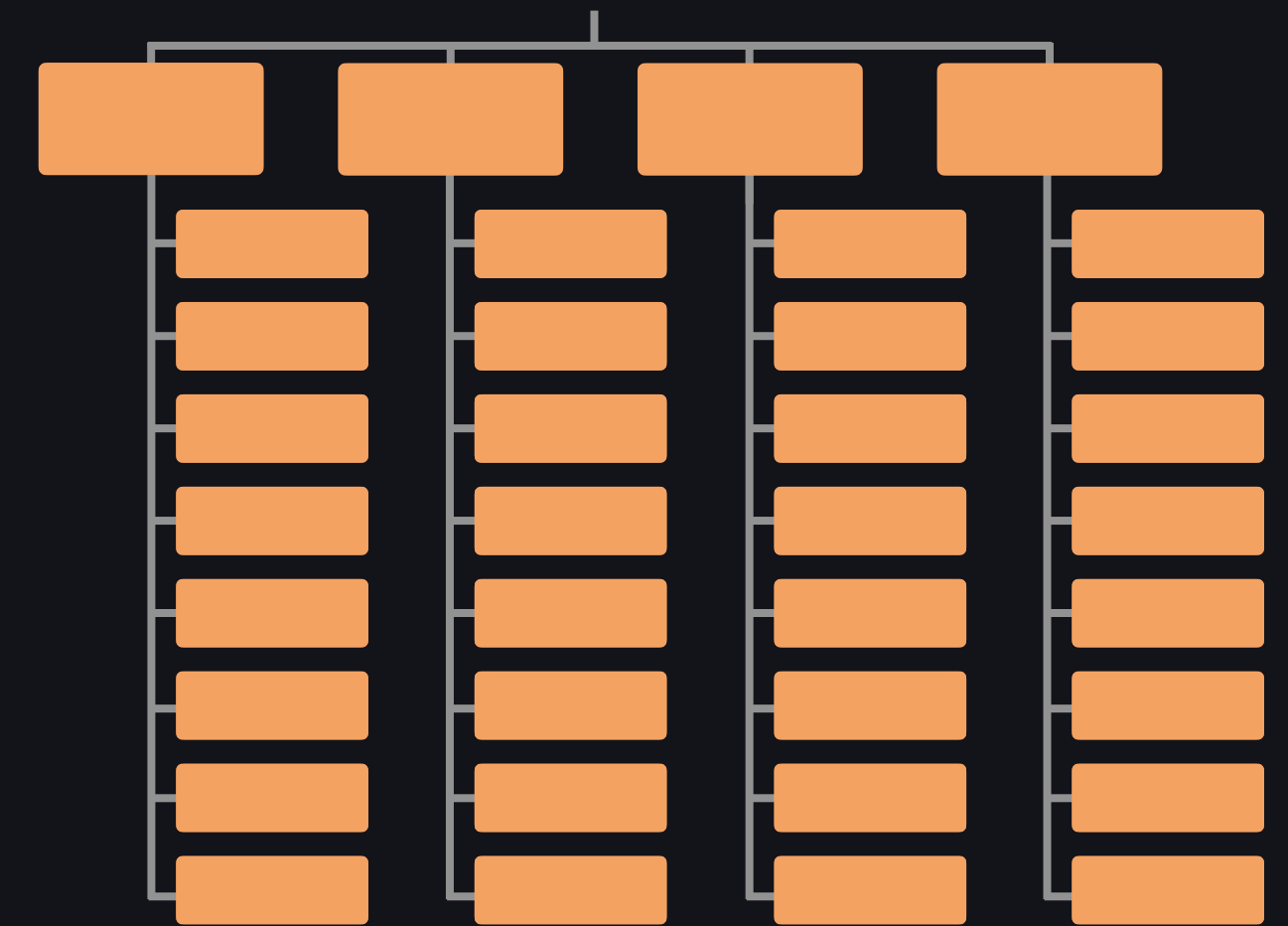
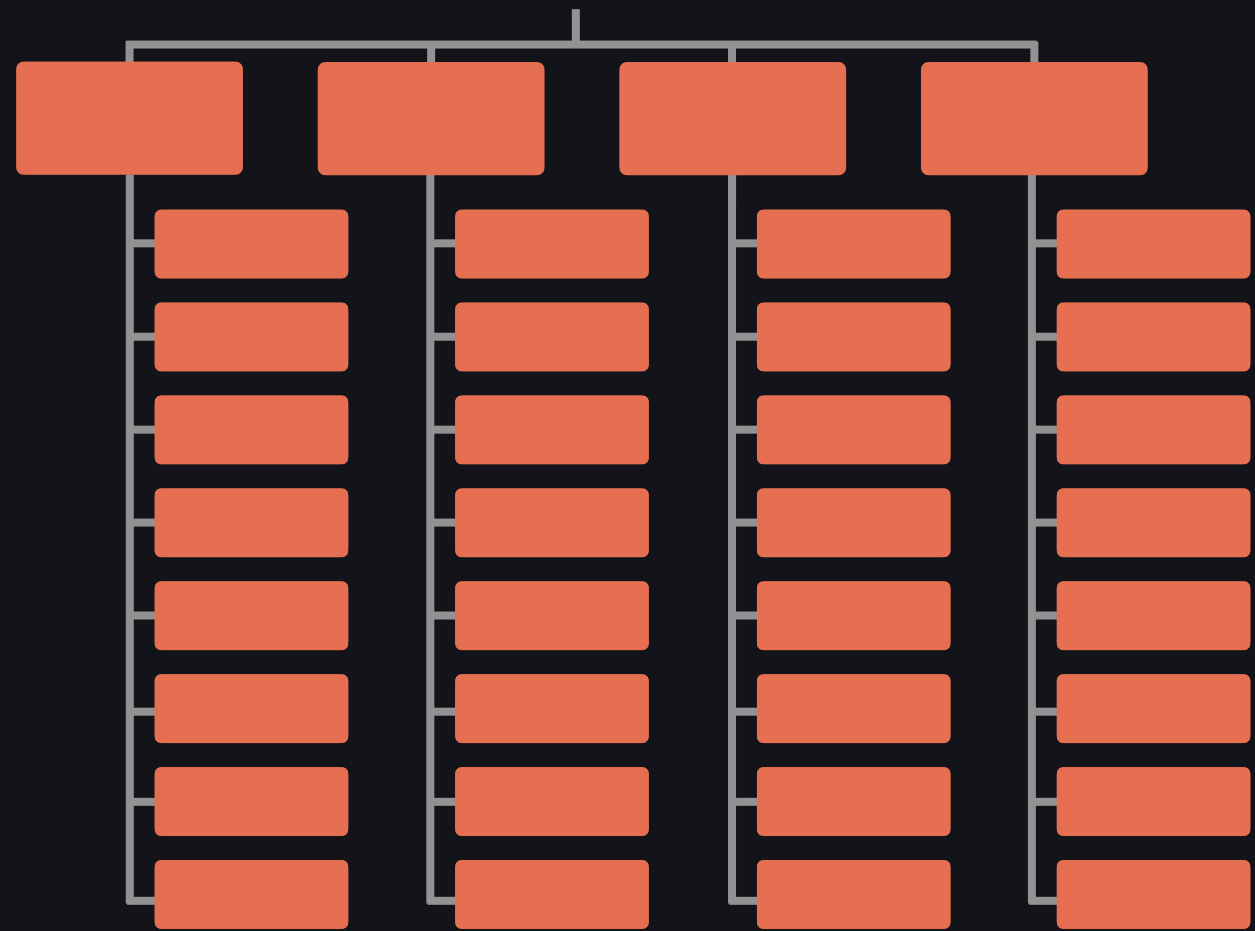
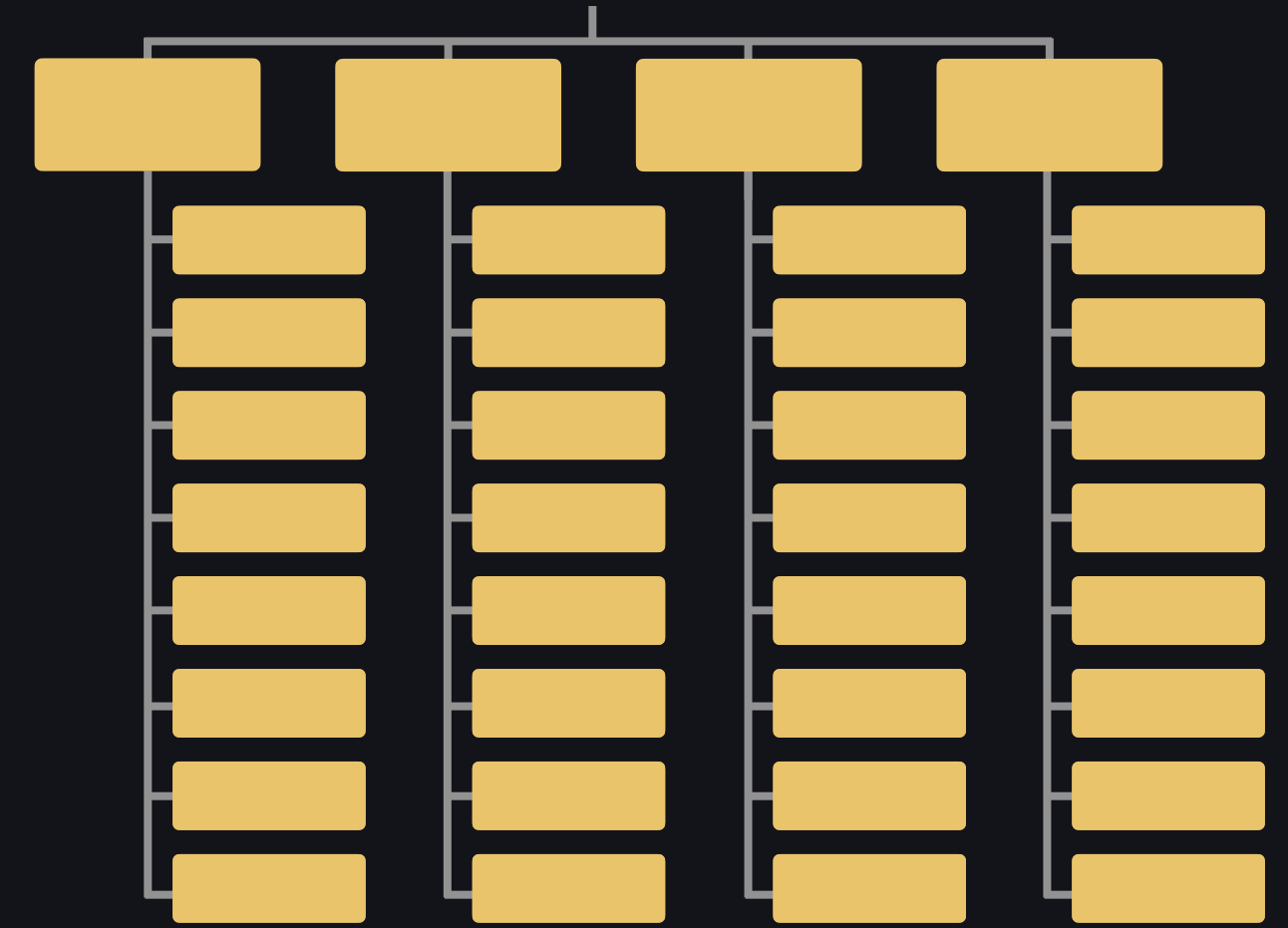




what could these do



to set the teams up for success?

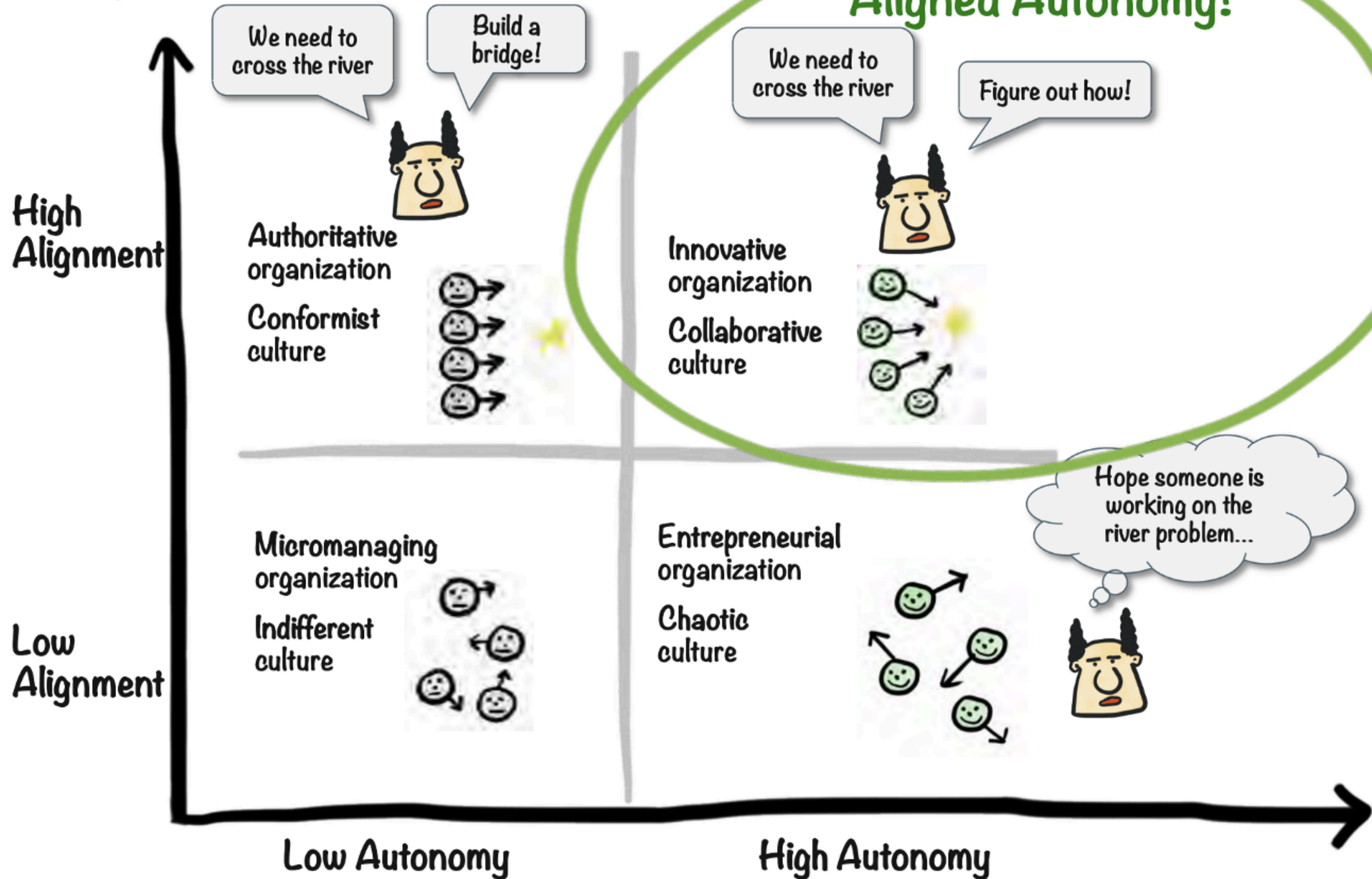




ALIGNMENT

in the age of **autonomy**

Alignment enables Autonomy





principles on the company wiki

ISN'T GOOD ENOUGH!





for (**team** in teams) {

with **RESPECT** for the **team** autonomy

make the **team** **WANT** to do it

and provide **HELPFUL GUIDANCE**

adapted to their **CONTEXT** and **ABILITIES**

without exceeding their **COGNITIVE CAPACITY**

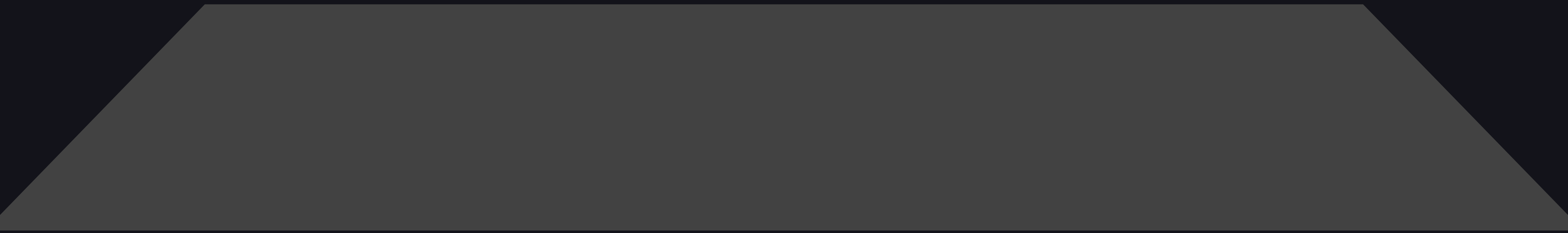
}



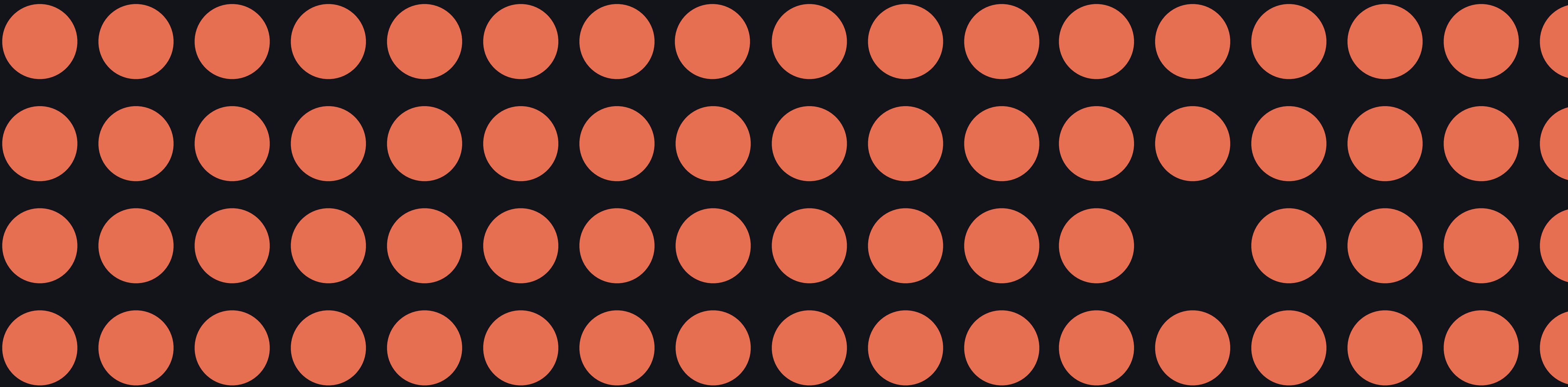
technology **RADAR**



weekly technical DEMO



weekly technical DEMO



technical **DIRECTION**



Technical
direction

Audun F. Strand Truls Jørgensen



Technical direction

Audun F. Strand Truls Jørgensen

n on a data platform

Hybr

ption layer

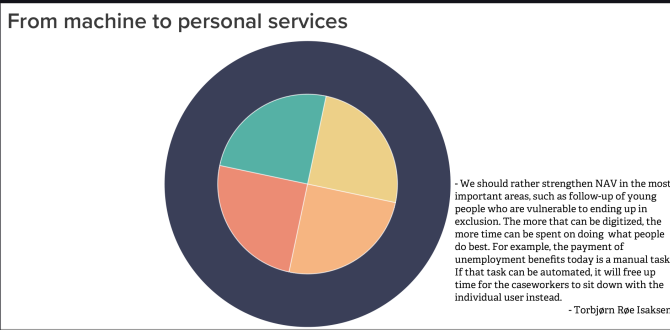
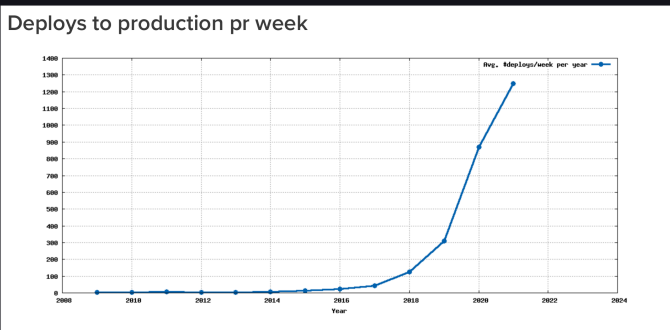
Centralised

Technical direction

y radar

Audun F. Strand Truls Jørgensen

System centri



Our plan:

Present this for all teams in NAV
Discuss how it affects your team

We can

- discuss in more detail in a follow up meeting
- provide training material / background material for each topic
- provide subject matter experts to help on a specific topic

If you want.

How to read this presentation:

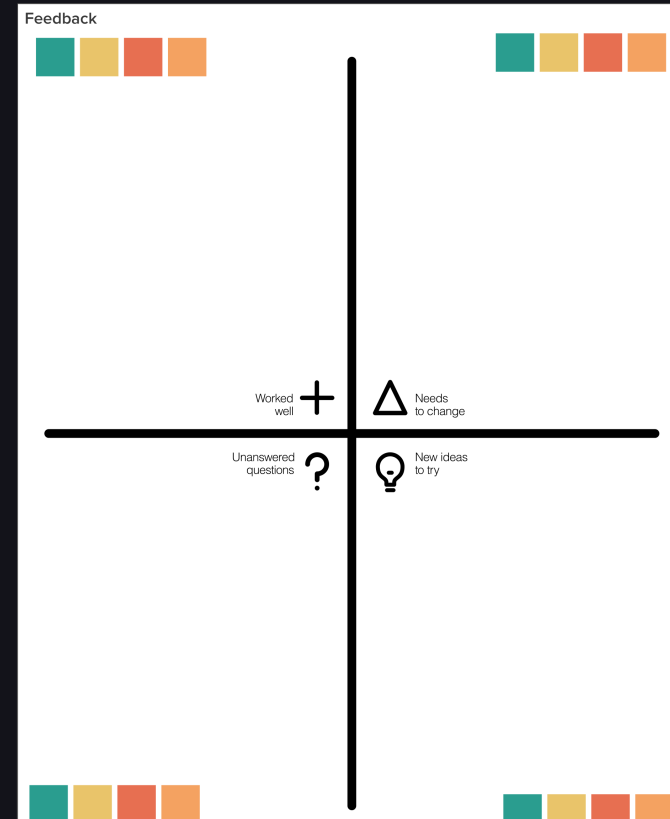
Each topic has a set of arrows, where each arrow indicates a direction from something and to something

NAV is heterogeneous. A team's ability to change differs a lot. It depends on not only the level of internal technical ownership, but on how adaptive to change their software systems are.

Our systems has different starting points and different ability to change, but we encourage every team to try to move

From _____ To _____

#teknisk_reining



Development

- Feature branching
- Trunk based development
- CI/CD

Deployment

- Blue/green
- Canary
- Rollback

Monitoring

- APM
- Logging
- Alerting

Configuration

- GitOps
- Infrastructure as Code

Security

- Penetration testing
- Vulnerability scanning

Compliance

- Audit logs
- Access control

Performance

- Load testing
- Capacity planning

Documentation

- API docs
- Architecture diagrams

Accessibility

- WCAG
- Screen reader support

Localization

- i18n
- l10n

Logging

- Structured logging
- Log aggregation

Testing

- Unit tests
- Integration tests

Tools

- CI/CD
- Containerization
- Cloud providers

Data driven

- Share data with REST/SOAP/DB → Share data as streams/ versioned data sets
- Point-to-point integration → Publish events
- Monitor the applications → Monitor the user behaviour
- Analytic data as by-product → Data products in product areas
- Hard to find data → Data mesh on a data platform
- Dependency on data models we don't own → Anti corruption layer

Techniques

- Feature Branch → Trunk based development
- Shared test environments → Isolated component and E2E-tests
- List of 136 compliance requirements → Built in compliance
- Ad hoc handling of technical debt → At least 25% of team capacity on technical maintenance
- Hybrid cloud → Multi cloud
- Centralised security → Distributed, built in security Security Champions in teams

Culture

- Centralized governance of approved technologies → Technology radar
- 10x developer → Pair programming
- Closed source code → Code in the open
- Remote friendly → Remote first
- Input/Output → Outcome
- Over time pizza → Team health checks

Sosioteknisk

- System centric architecture → Sosioteknisk architecture
- Horizontal, layered responsibility → Teams responsible for vertical
- Centralized, coordinated interaction → Team api
- Team → Team topologies
- Product areas backed by financing programs → Stable financing of product areas
- Projects in financing programs → Domain driven team organization

Technical direction

Audun F. Strand, Truls Jørgensen

Team Based Development

- Self-organizing teams
- Cross-functional teams

Product Development

- Product Backlog
- Sprint cycles

Deployment Automation

- Infrastructure as Code
- CI/CD Pipelines

Security Automation

- Automated vulnerability scanning
- Automated penetration testing

Multi-cloud

- Cloud providers
- Cloud management

Compliance Automation

- Automated audits
- Policy as Code

Performance

- Load testing
- Capacity planning

Documentation

- API docs
- Architecture diagrams

Accessibility

- WCAG
- Screen reader support

Localization

- i18n
- l10n

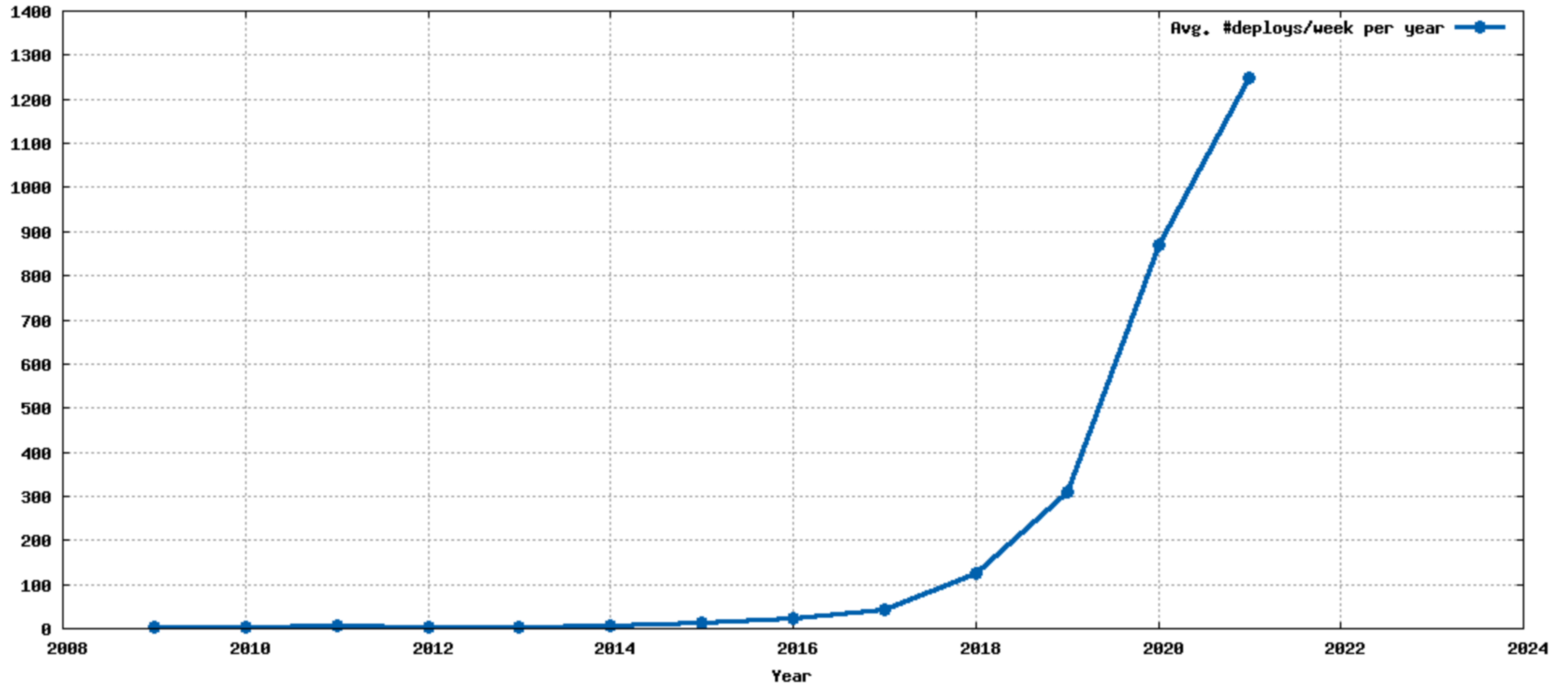
Logging

- Structured logging
- Log aggregation

Testing

- Unit tests
- Integration tests

Deploys to production pr week



Our plan:

Present this for all teams in NAV
Discuss how it affects your team

We can

- discuss in more detail in a follow up meeting
- provide training material / background material for each topic
- provide subject matter experts to help on a specific topic


If you want.

How to read this presentation:

Each topic has a set of arrows, where each arrow indicates a direction from something and to something

NAV is heterogeneous. A team's ability to change differs a lot. It depends on not only the level of internal technical ownership, but on how adaptive to change their software systems are.

Our systems has different starting points and different ability to change, but we encourage every team to try to move

From  **To**

Open mural



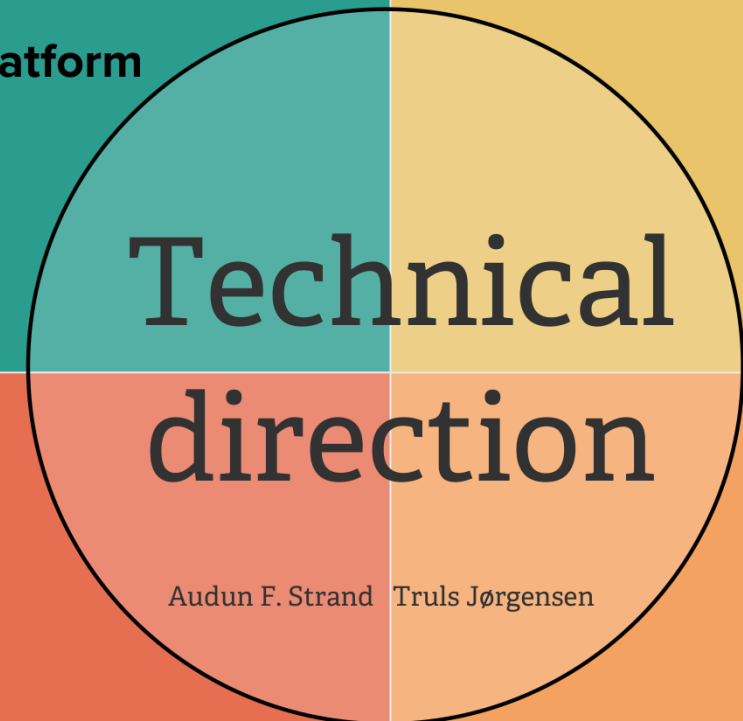
#teknisk_retning

Data driven

Share data with REST/SOAP/DB	→	Share data as streams/ versioned data sets
Point-to-point integration	→	Publish events
Monitor the applications	→	Monitor the user behaviour
Analytic data as by-product	→	Data products in product areas
Hard to find data	→	Data mesh on a data platform
Dependency on data models we don't own	→	Anti corruption layer

Techniques

Feature Branch	→	Trunk based development
Shared test environments	→	Isolated component and E2E-tests
List of 136 compliance requirements	→	Built in compliance
Ad hoc handling of technical debt	→	At least 25% of team capacity on technical maintenance
Hybrid cloud	→	Multi cloud
Centralised security	→	Distributed, built in security Security Champions in teams



Culture

Centralized governance of approved technologies	→	Technology radar
10x developer	→	Pair programming
Closed source code	→	Code in the open
Remote friendly	→	Remote first
Input/Output	→	Outcome
Over time pizza	→	Team health checks

Sociotechnical

System centric architecture	→	Sociotechnical architecture
Horizontal, layered responsibility	→	Teams responsible for vertical
Centralized, coordinated interaction	→	Team api
Team	→	Team topologies
Product areas backed by financing programs	→	Stable financing of product areas
Projects in financing programs	→	Domain driven team organization



Data driven

Share data with REST/SOAP/DB



Share data as streams/
versioned data sets

Point-to-point integration



Publish events

Monitor the applications



Monitor the user behaviour

Analytic data as by-product



Data products in product areas

Hard to find data



Data mesh on a data platform

Dependency on data models
we don't own



Anti corruption layer

Tech

♥ Culture

Centralized governance of approved technologies

10x developer

Closed source code

Remote friendly

Input/Output

Over time pizza



**BE
OPEN
AND USE
OPEN
SOURCE**

direc

Audun F. Strand



Techniques

Feature Branch



Trunk based development

Shared test environments



Isolated component and E2E-tests

List of 136 compliance requirements



Built in compliance

Ad hoc handling of technical debt



At least 25% of team capacity on technical maintenance

Hybrid cloud



Multi cloud

Centralised security



**Distributed, built in security
Security Champions in teams**

Technical

ction

Truls Jørgensen



Sosiotechnical

System centric architecture →

Sosiotechnical architecture

Horizontal, layered responsibility →

Teams responsible for vertical

Centralized, coordinated interaction →

Team api

Team →

Team topologies

**Product areas backed by
financing programs** →

Stable financing of product areas

Projects in financing programs →

Domain driven team organization

Data driven

Share data with REST/SOAP/DB



Share data as streams/
versioned data sets

Point-to-point integration



Publish events

Monitor the applications



Monitor the user behaviour

Analytic data as by-product



Data products in product areas

Hard to find data



Data mesh on a data platform

Dependency on data models
we don't own



Anti corruption layer

Techniques

Feature Branch



Trunk based development

Shared test environments



Isolated component and E2E-tests

List of 136 compliance requirements



Built in compliance

Ad hoc handling of technical debt



At least 25% of team capacity on
technical maintenance

Hybrid cloud



Multi cloud

Centralised security



Distributed, built in security
Security Champions in teams

Technical
direction

Audun F. Strand Truls Jørgensen

Culture

Centralized governance of
approved technologies



Technology radar

10x developer



Pair programming

Closed source code



Code in the open

Remote friendly



Remote first

Input/Output



Outcome

Over time pizza



Team health checks



Sosioteknisk

System centric architecture



Sosioteknisk architecture

Horizontal, layered responsibility



Teams responsible for vertical

Centralized, coordinated interaction



Team api

Team



Team topologies

Product areas backed by
financing programs

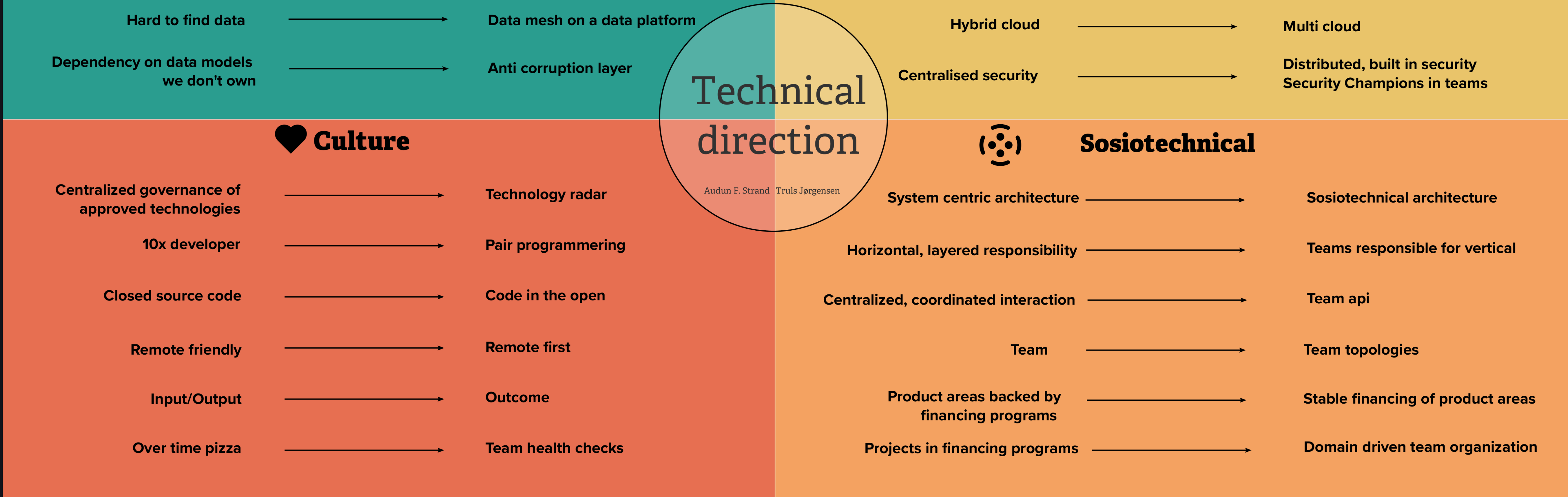


Stable financing of product areas

Projects in financing programs



Domain driven team organization





GOLDEN PATH platform



CAKE driven development



CAKE driven development



CAKE driven development



CAKE driven development



CAKE driven development



what we **adopt** and **prefer** when optimising for

SPEED AND FAST FLOW

in Norway's largest bureaucracy



with **RESPECT** for the team autonomy

make the team **WANT** to do it

and provide **HELPFUL GUIDANCE**

adapted to their **CONTEXT** and **ABILITIES**

without exceeding their **COGNITIVE CAPACITY**





optimising for

FAST FLOW

in Norway's largest bureaucracy





THANK YOU!

@audunstrand @trulsjor

