

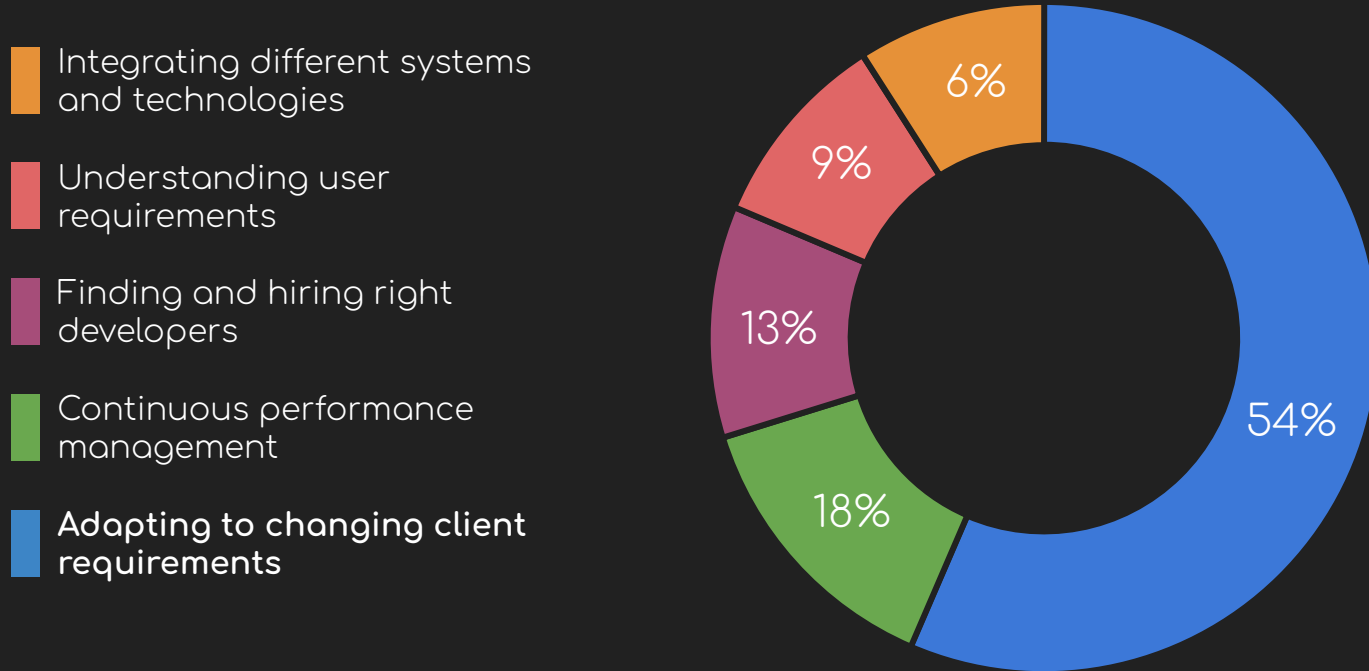
# How To Build Future-Proofed Applications



Masking Technology  
hello@masking.tech  
[linkedin.com/company/maskingtechnology](https://www.linkedin.com/company/maskingtechnology)



# Biggest challenges of software development



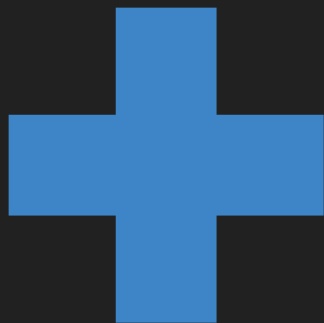
(GoodFirms. (2023, October 18). Remarkably Useful Stats and Trends on Software Development)

# Future-Proofed Applications

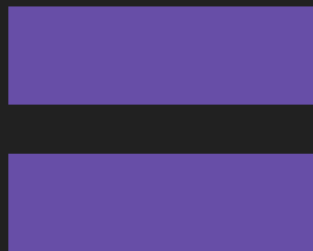
---

Adaptable, without heavy refactoring  
and breaking beyond repair.

- Wikipeter -



Extendable



Modifiable



Reducible

Time & cost



Productivity





#1 Requirements

#2 Design

#3 Realisation

#4 Verification

#5 Maintenance

#1

Keep it Small

Requirements





LESS is MORE

# BACKLOG

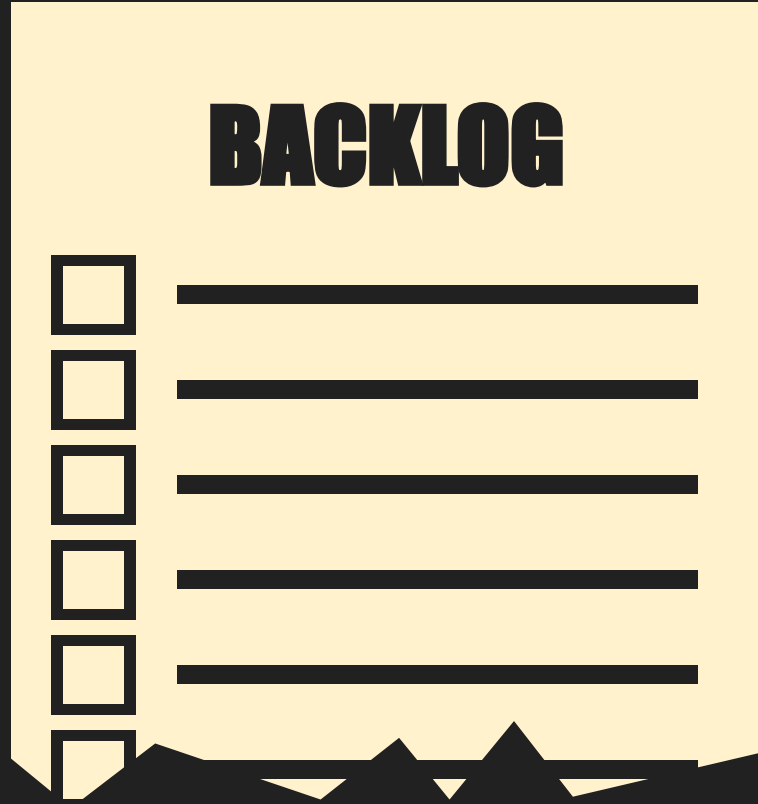
\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_



▶ Must haves

~~Should haves~~

~~Could haves~~

~~Would haves~~

# Acceptance criteria



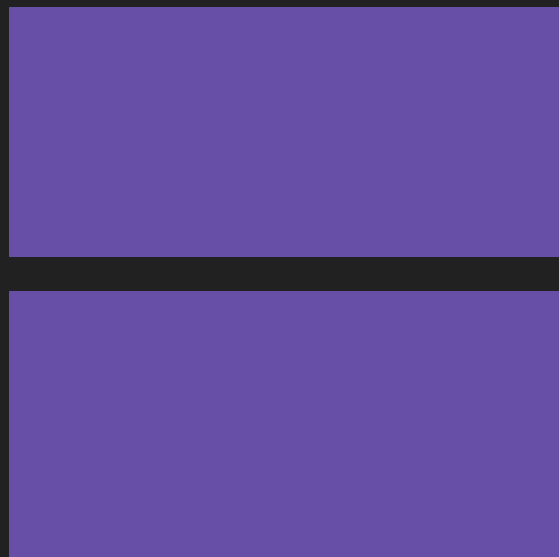
- Understandable
- Value > Effort
- Value > Risk
- Stable

# #2

## Keep it Simple

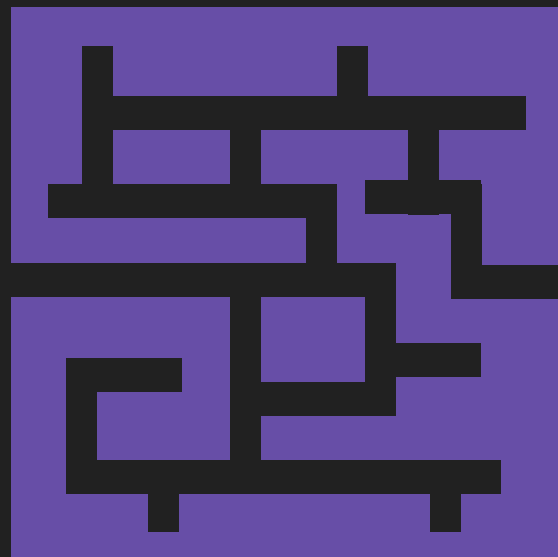
Design





Simple

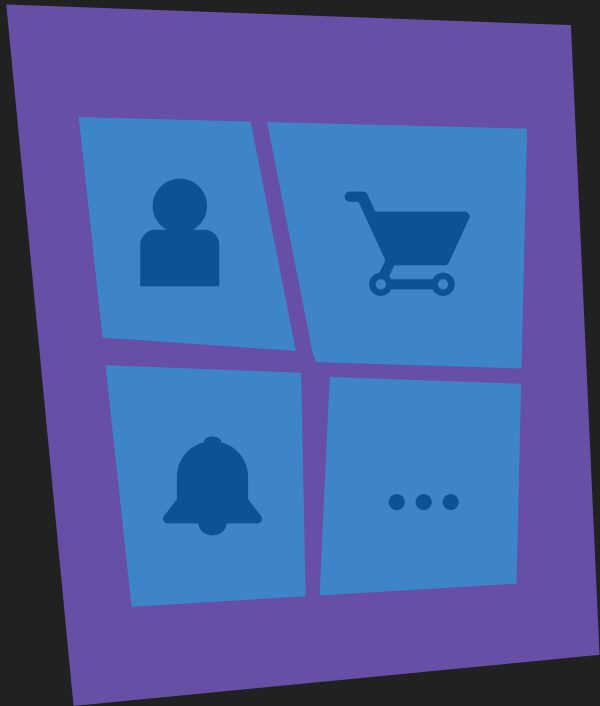
VS



Complex



Good  
architecture  
is fundamental

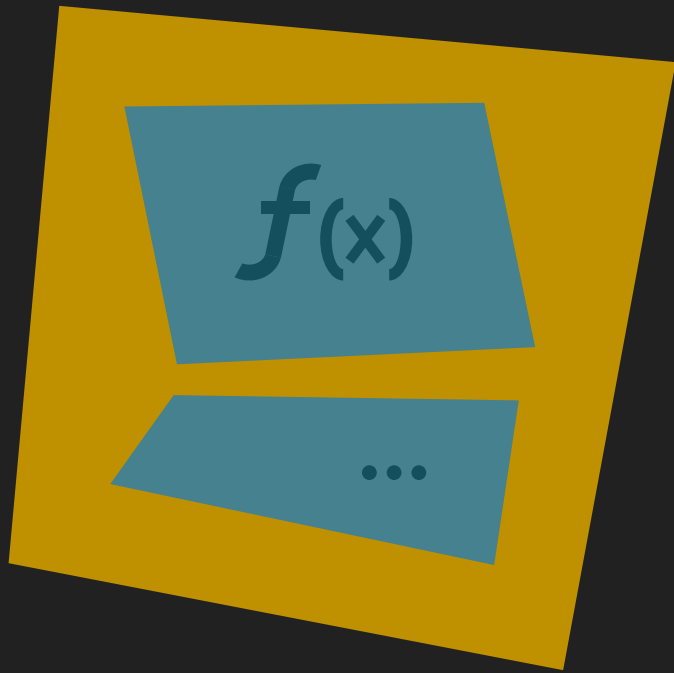


Strong  
modularization  
is crucial



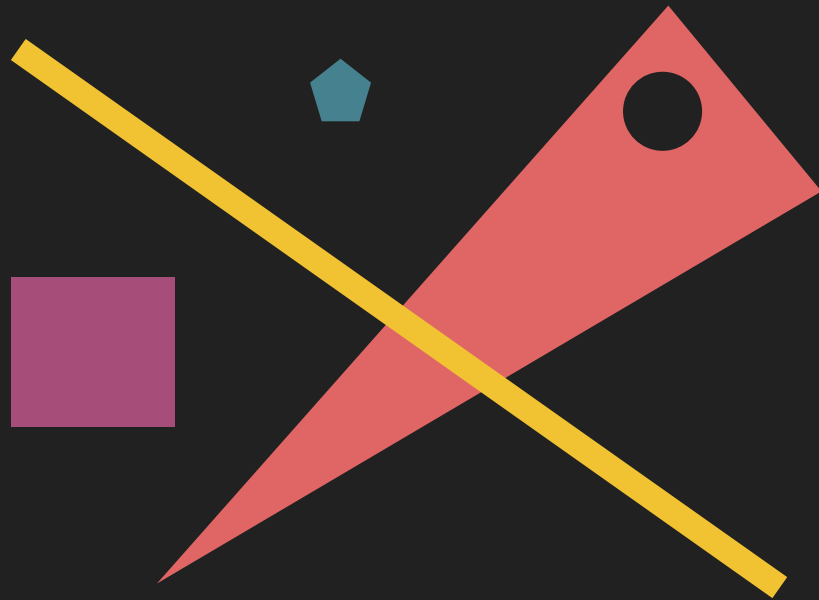


Strong  
componentization  
is essential

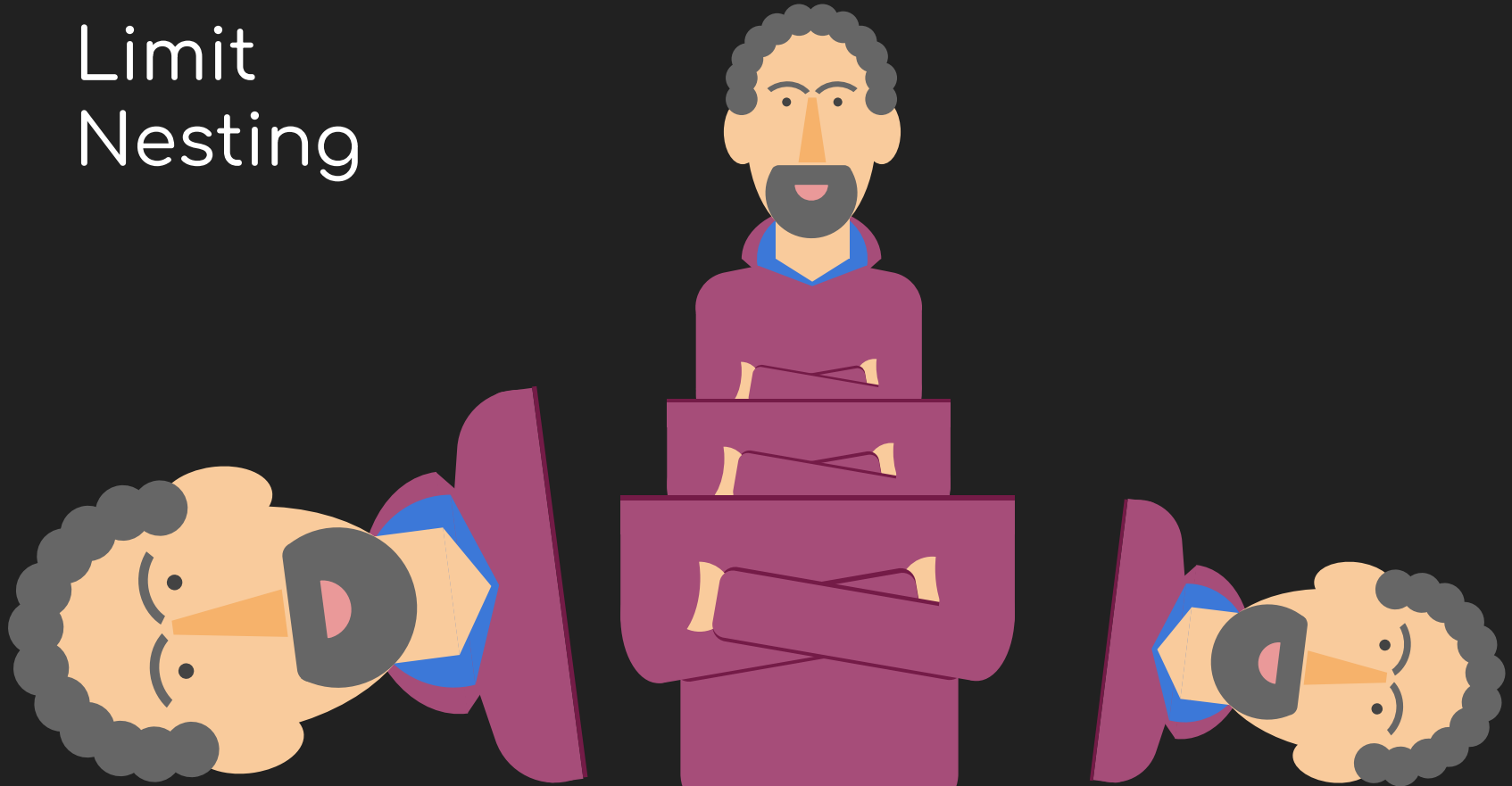


Strong  
encapsulation  
is vital

# Limit Abstractions



# Limit Nesting



#3

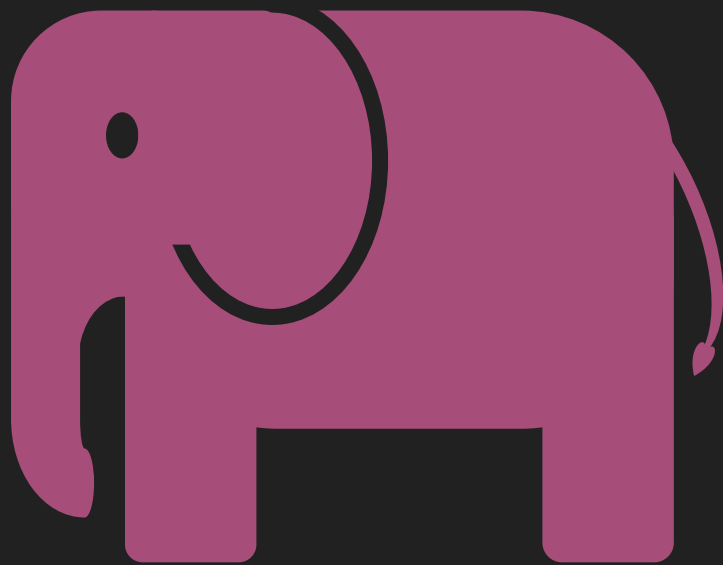
Keep it Elegant

Realisation



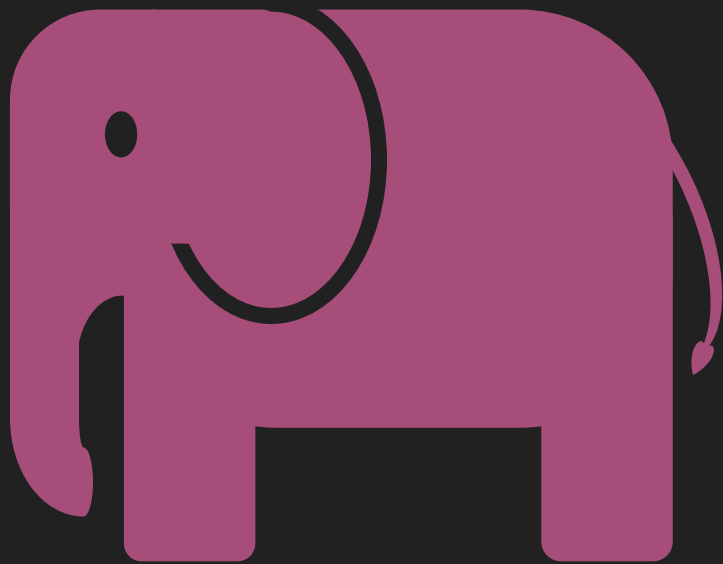
Code is ~~poetry~~  
documentation

// This is a slide

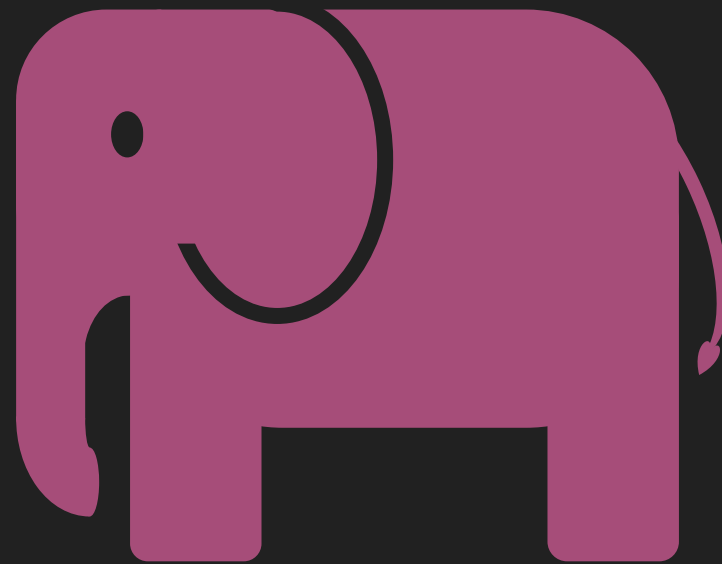


Dining table





Dining table



Hair dryer

629

78

6



402

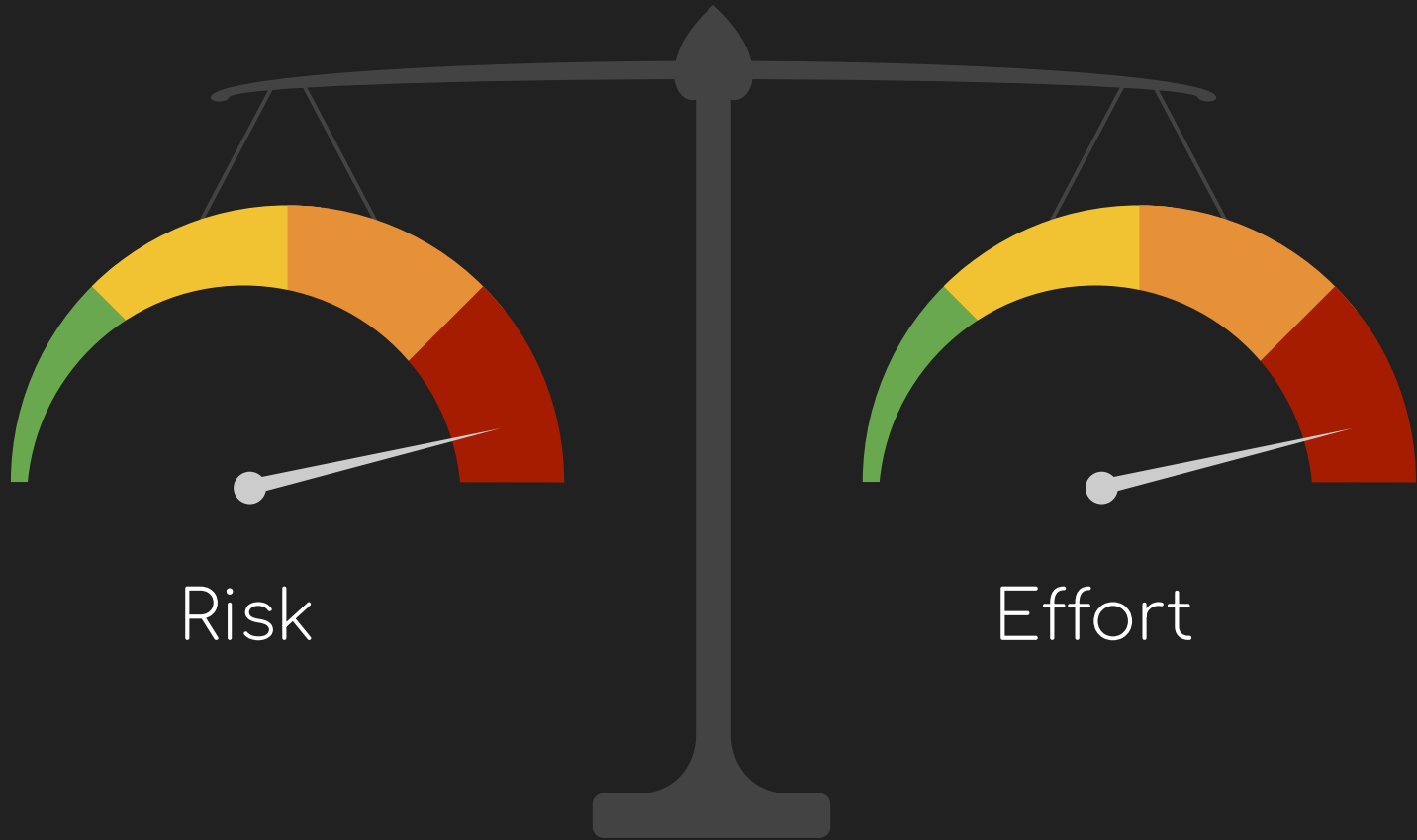
32

# #4

## Keep it Safe

Verification





Risk

Effort



Use case

Misuse case

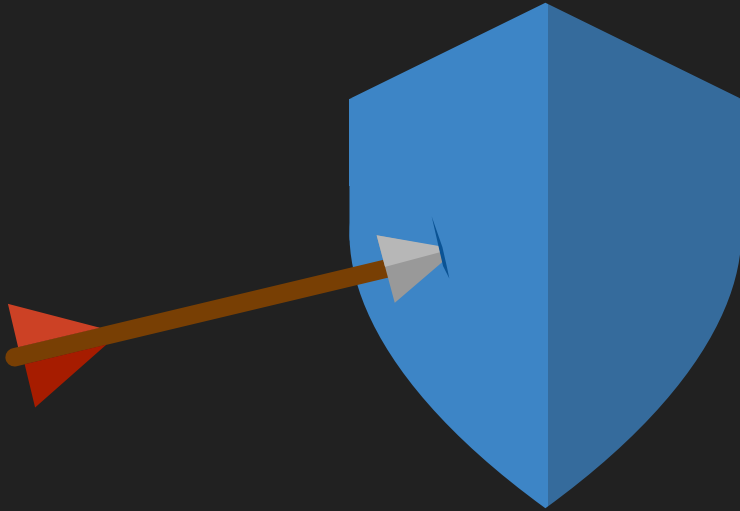


# Ensure correct operation



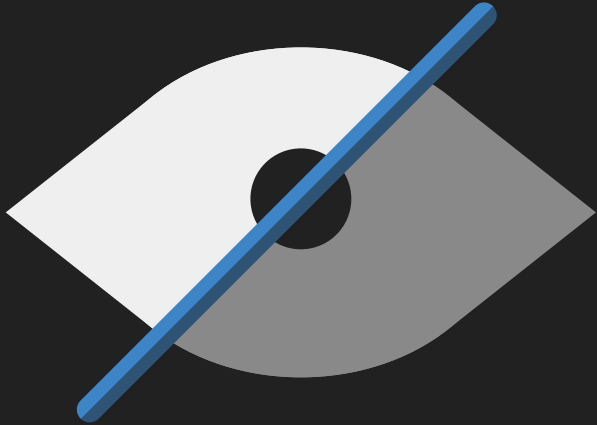
- User Interface
- Use Cases
- Integrations

# Ensure security measures



- Access Control
- Content Validation
- Traceability

# Ensure privacy measures



- Agreements
- Storage location
- Data Encryption

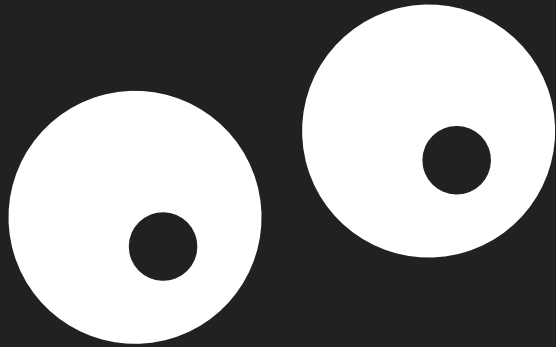


# Ensure market value



- User Feedback
- Competitive Analysis
- Regulatory Compliance

# Ensure development continuity



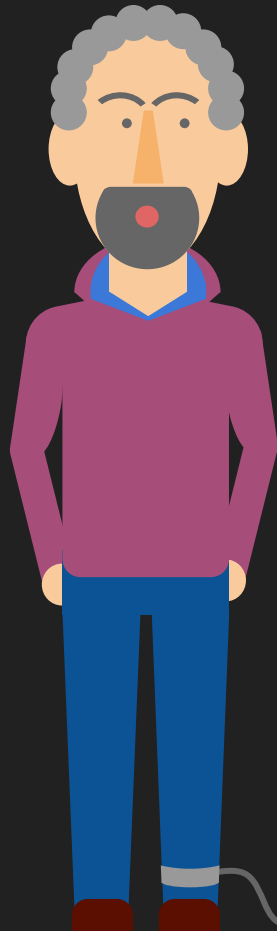
- Work in pairs
- Review all work

#5

Keep it Updated

Maintenance





DEBT

#1 Requirements

Feedback, compliance, etc.

#2 Design

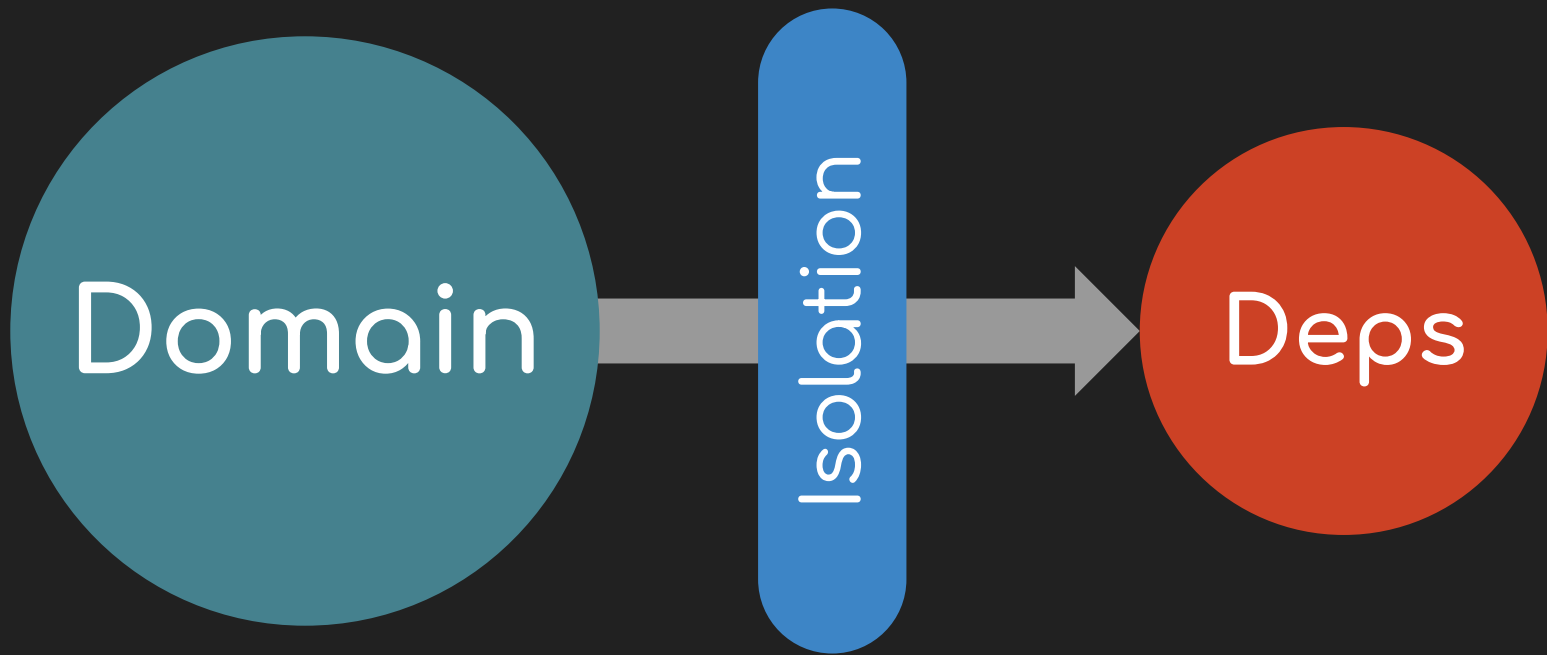
Schemes, documents, etc.

#3 Realisation

Language & dependencies.

#4 Verification

Tests, market value, etc.



Domain

Isolation

Deps

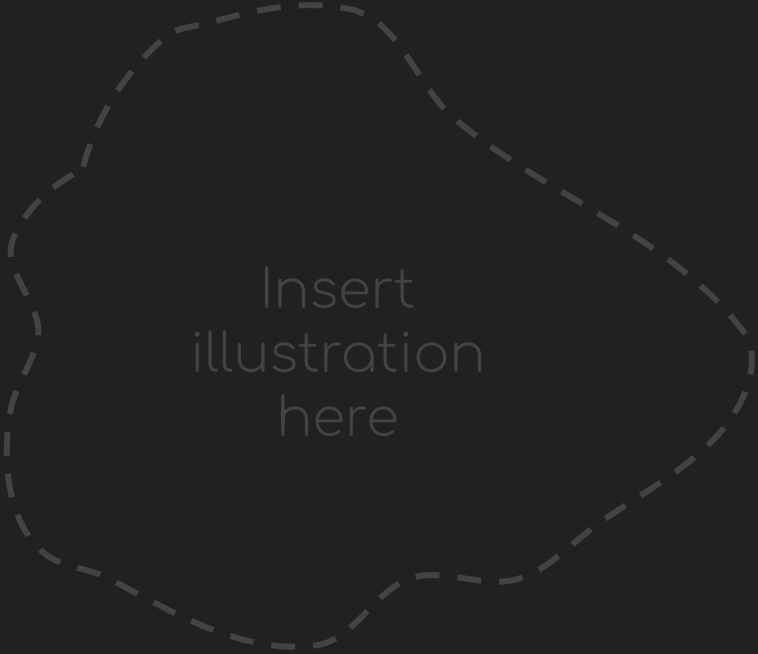


PLEASE  
REMOVE  
OBSOLETE  
FEATURES

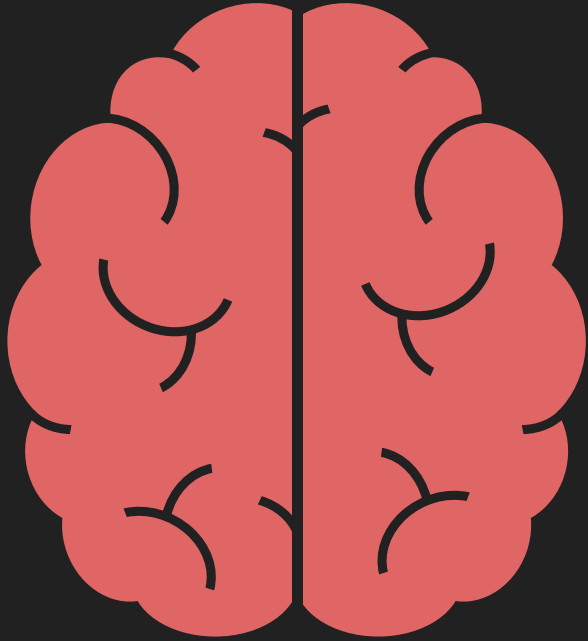
How to apply



Adaptability  
requires  
**discipline**

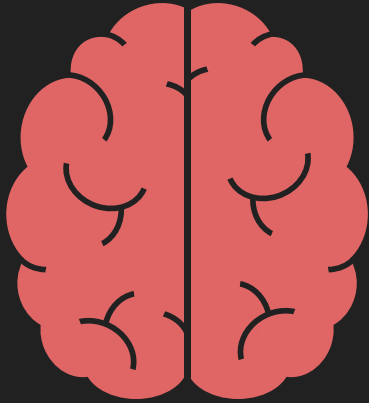


Insert  
illustration  
here

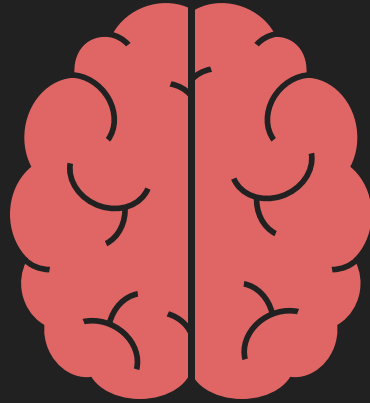


But also a  
**growth mindset**

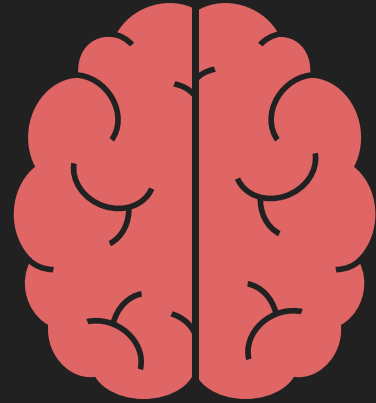
Alignment



Users



Business

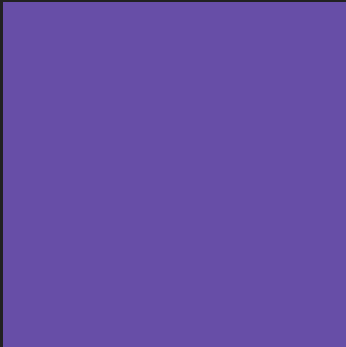


Development

Key takeaways

Can I  
join?

Should have

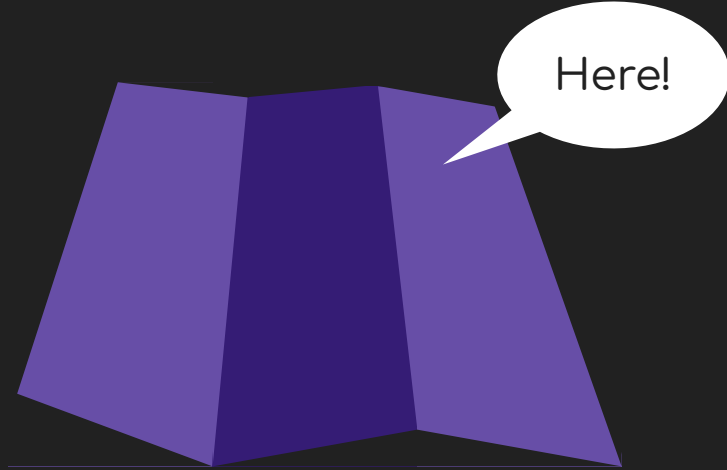


NO!

Strong requirement selection

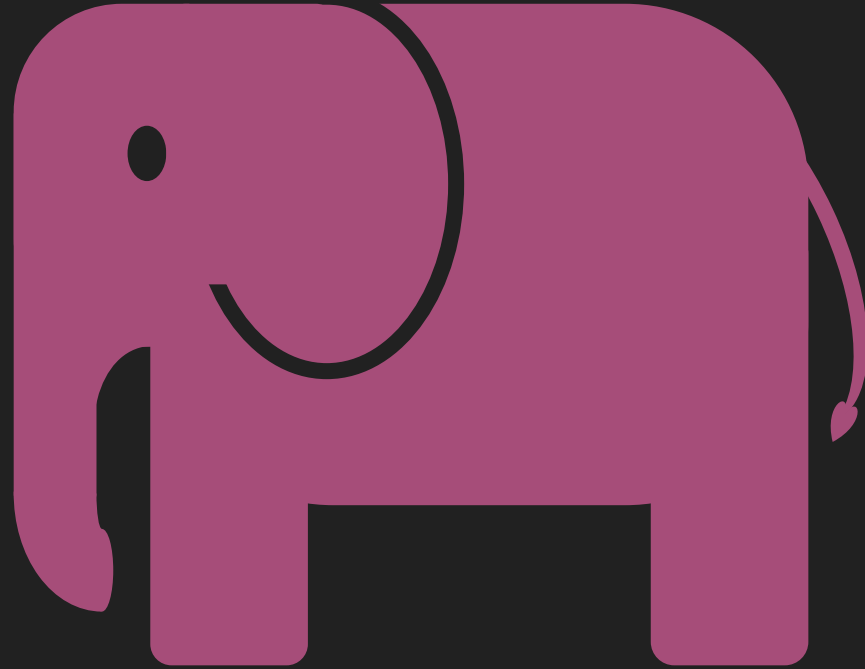
Where do I go?

Must-have



Navigable application architecture

You know  
what I am  
and do.



Self explanatory code

Everything  
is under  
control!

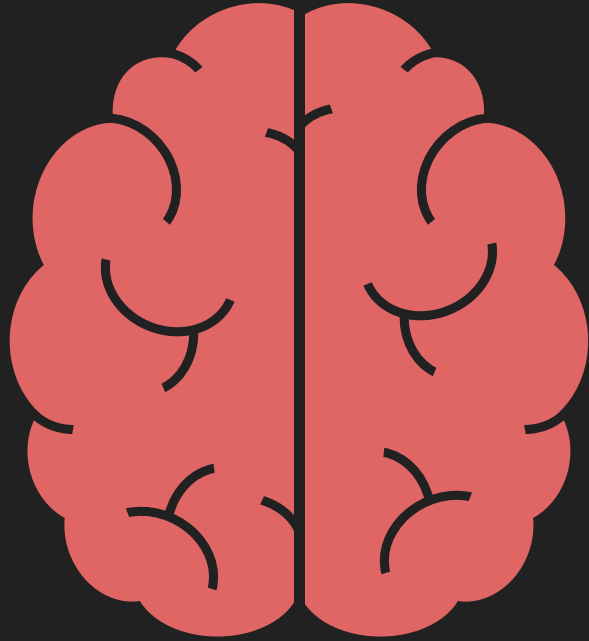


Risks mitigated





Debt cleared



Keep  
thinking

# Thanks!



<https://masking.tech>

Bas Meeuwissen

[bas@masking.tech](mailto:bas@masking.tech)

[linkedin.com/in/basmeeuwissen](https://www.linkedin.com/in/basmeeuwissen)



Peter van Vliet

[peter@masking.tech](mailto:peter@masking.tech)

[linkedin.com/in/petervliet](https://www.linkedin.com/in/petervliet)

