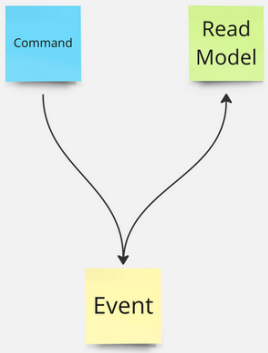
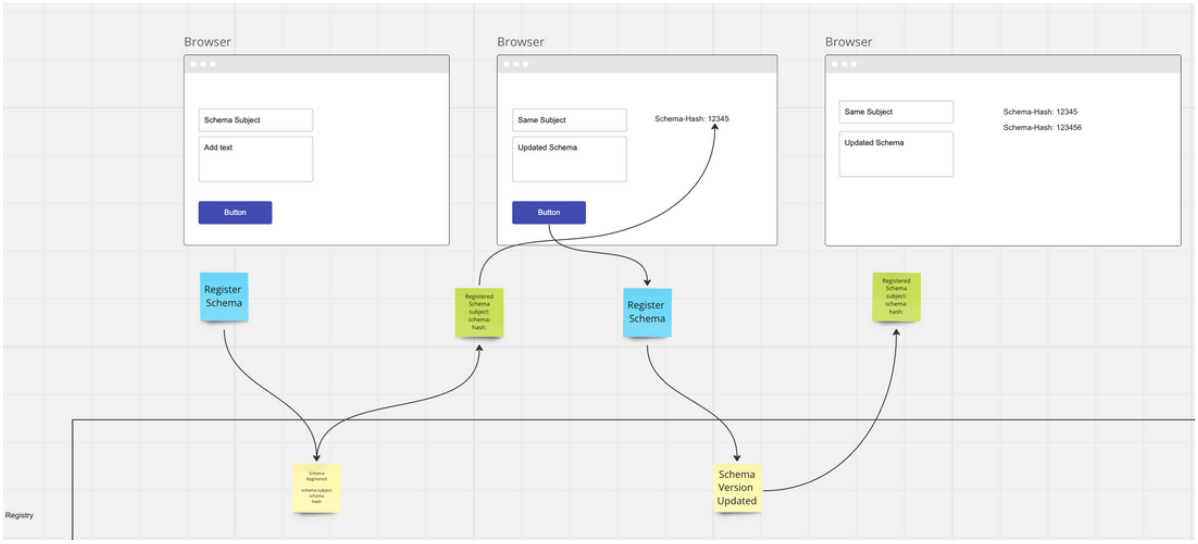
NEBULIT

SOFTWARE OHNE SCHÄTZUNG.

IN 8 SCHRITTEN ZU BESSEREN SOFTWARE REQUIREMENTS MIT EVENTMODELING



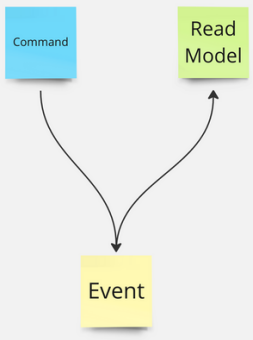


Willst du mehr über das Thema erfahren?
Folge mir auf LinkedIn

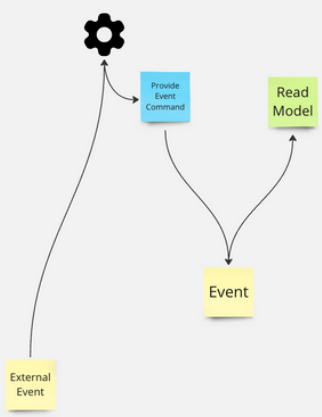
Eventmodeling bietet dir und deinem Team eine einheitliche Sprache zur Erfassung und Dokumentation von Anforderungen an ein Softwaresystem.

Das Eventmodel kann gleichzeitig als Grundlage für die Umsetzung dienen. Die einzelnen Schritte hierfür findet du in diesem Dokument.

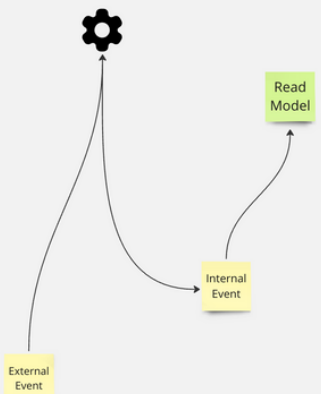
Interaktion



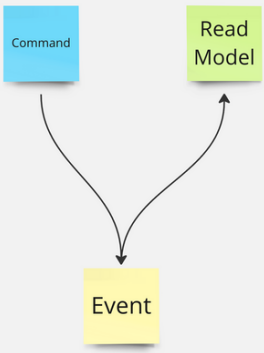
Automatisierung

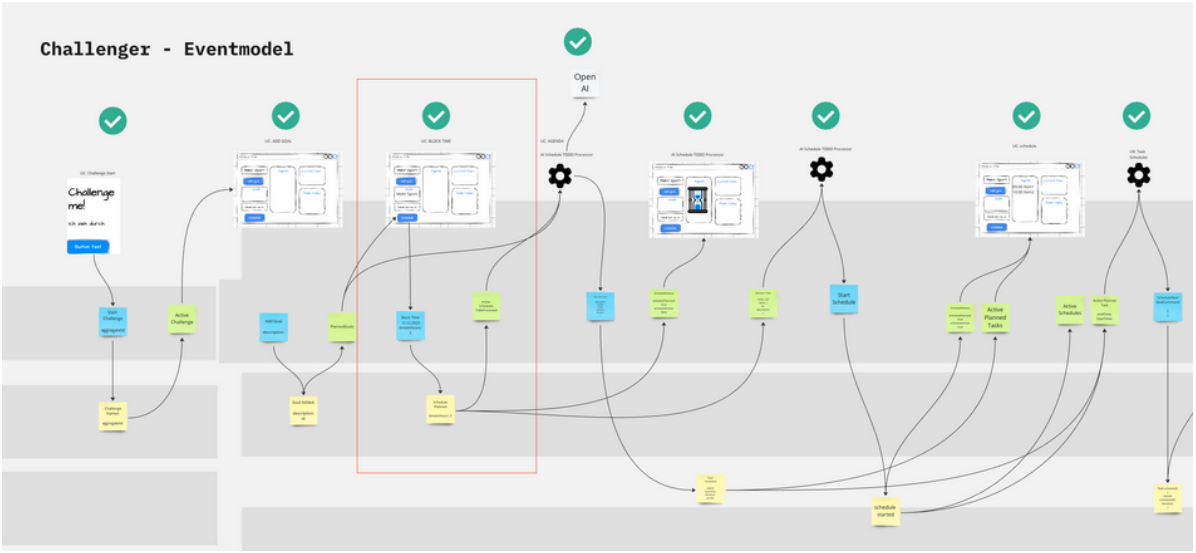


Übersetzung



BEVOR ES LOSGEHT.





Regeln

Ein Eventmodell zu erstellen ist nicht schwer. Damit du aber alle Vorteile bekommst, gibt es ein paar Regeln, an die du dich halten solltest.

Das Eventmodell bildet eure Software entlang einer Zeitachse ab.

Pfeile gehen daher immer nur nach vorne (keine Zeitsprünge zurück)

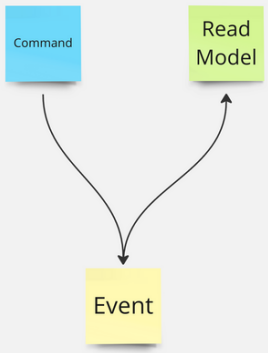
Pro Abschnitt im Eventmodell (Slice) wird immer nur eine Aktion abgebildet.

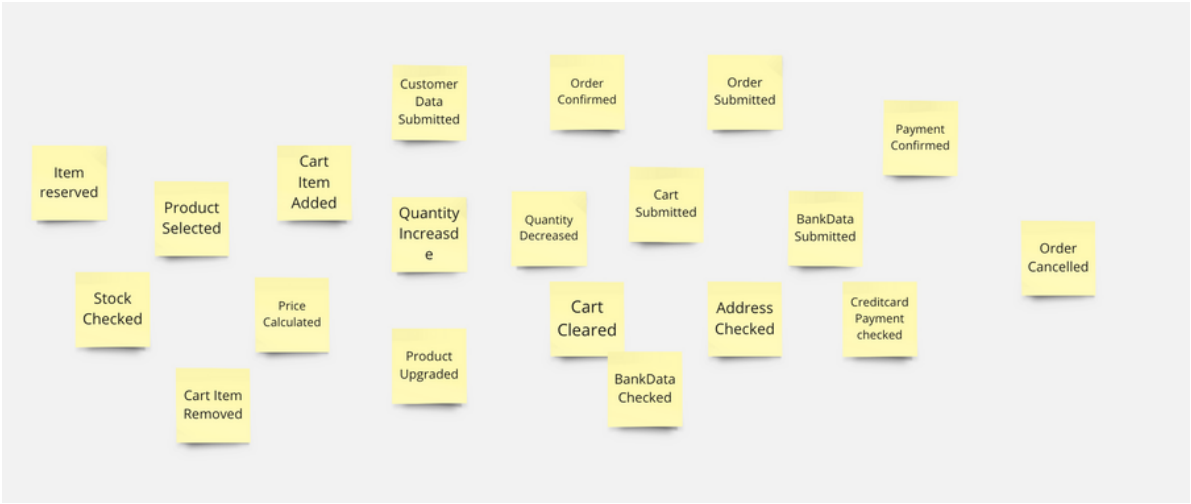
Es gibt keine parallelen Zeitstränge. Das Eventmodell liest sich von links nach rechts.

Es sind nur 8 Schritte.

Und jetzt.. viel Erfolg!

SCHRITT 1 -
ERFASSE DIE EVENTS





Wer?

Im ersten Schritt benötigt ihr so viele Teilnehmer wie möglich.
Es sollte eine Person geben, die den Workshop moderiert.

Teilnehmer:
Fachbereich, QS, Security, UX, Entwicklung, CTO, CEO.

Was?

Ihr erfasst sämtliche "Events", die im Softwaresystem auftreten können. Die Ereignisse sind in der Regel die Dinge, die jemand erwähnen würde, um das System zu beschreiben.

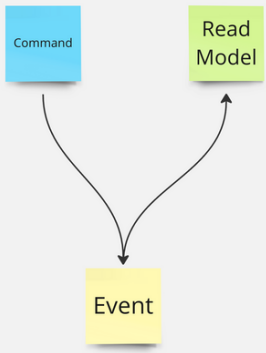
Wir formulieren Events in der **Vergangenheitsform**.

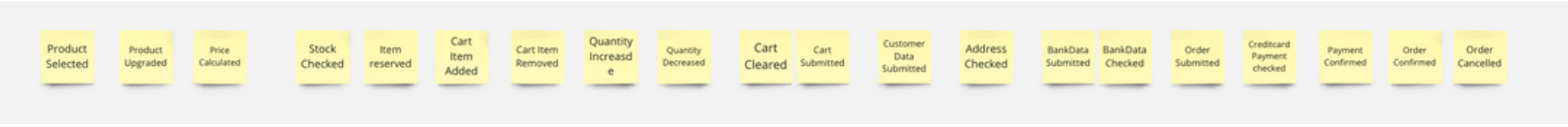
Wo?

Online: Miro

Vor Ort: großes Whiteboard bzw. Endlospapier

SCHRITT 2 –
SCHREIBE DIE STORY





Was?

Im nächsten Schritt bringt ihr die Events in die richtige Reihenfolge. Es ensteht die “**Story**”.

Es empfiehlt sich, die Events zunächst fachlich zu gruppieren und anschließend zu sortieren.

Ihr entfernt Duplikate und Events die keine sind. Dann bringt ihr alle Events in Form (Sprache, Vergangenheitsform)

Am Ende sind alle Events in einer Timeline.

Während der Sortierung können fehlende Events direkt ergänzt werden.

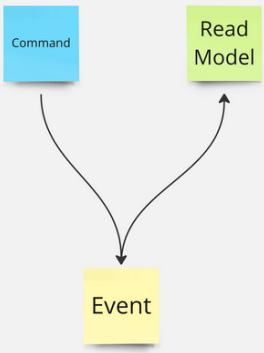
Wer?

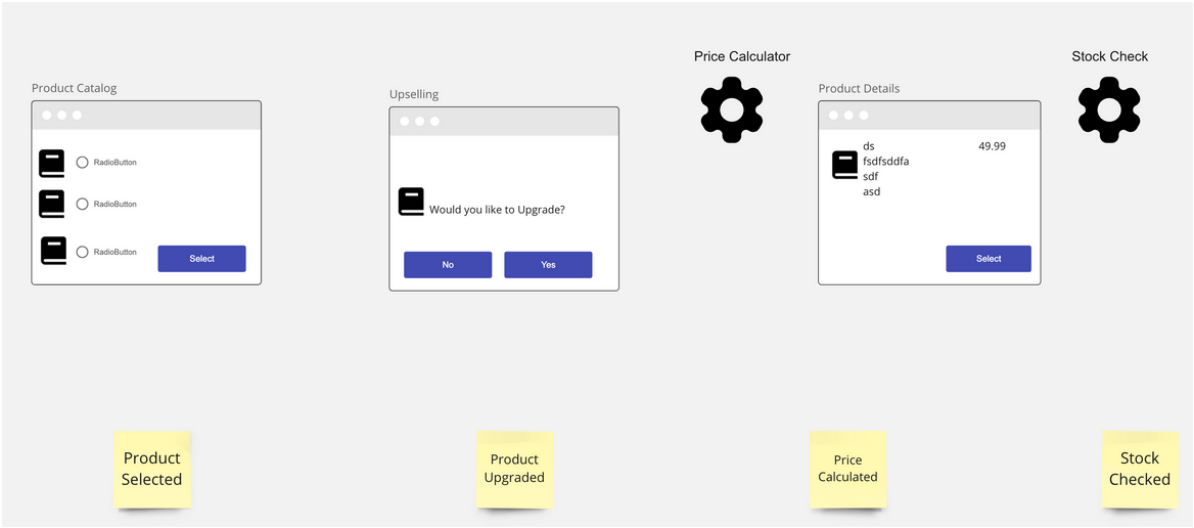
Sobald alle Events sortiert sind, liest eine freiwillige Person alle Events von links nach rechts laut vor und versucht eine konsistente Story zu erzählen.

Auch hier werden offensichtlich fehlende Events direkt ergänzt.

“Ab jetzt habt ihr einen ersten Überblick über die Größe des Systems”

SCHRITT 3 –
BESCHREIBE DIE SCREENS





Was?

In diesem Schritt beschreibt ihr die Screens für die einzelnen Schritte im System.

Euer Ziel ist es, die “Story” zu visualisieren.

Jede Aktion die ausgeführt wird hat einen eigenen Screen.
Werden mehrere Aktionen vom selben Screen gestartet, kann der Screen einfach dupliziert werden.

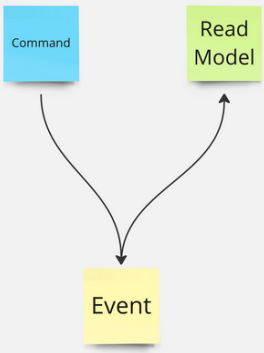
Wer?

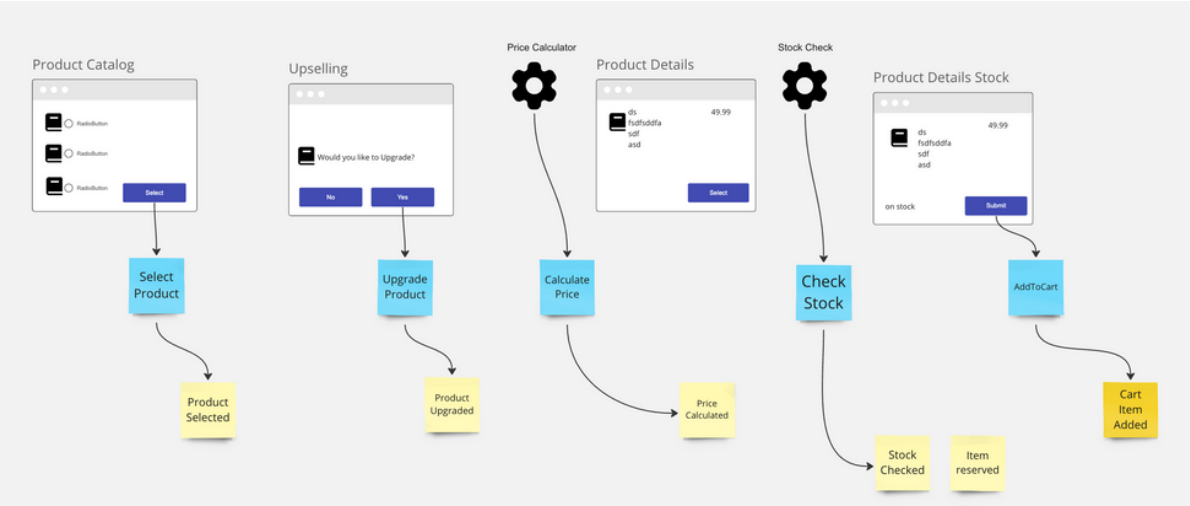
Dieser Schritt benötigt die meiste Zeit in der Vorbereitung. Es bietet sich an hier einen Break zu machen und zumindest die grundsätzlichen Screens in einer kleineren Gruppe vorzubereiten.

Screens müssen nicht detailliert und perfekt sein. Man sollte aber die Elemente erkennen können, die Aktionen ausführen und Daten visualisieren.

“Dieser Schritt kann von UX Seite bereits aus vorhandenen Screendesigns vorbereitet werden.”

SCHRITT 4 –
WELCHE AKTIONEN?



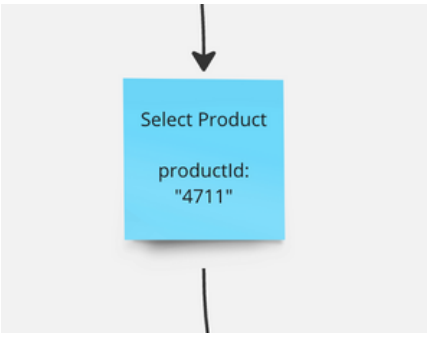


Was?

Dieser Schritt ist sehr wichtig.

Ihr definiert hier, welche Aktionen (“Commands”) im System tatsächlich möglich sind.
Ihr verwendet blaue Post-Its dafür.

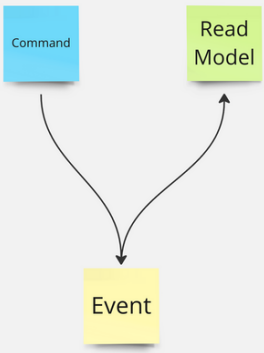
Die Commands werden so weit möglich bereits mit Beispiel-Daten angereichert. Je detaillierter desto besser!

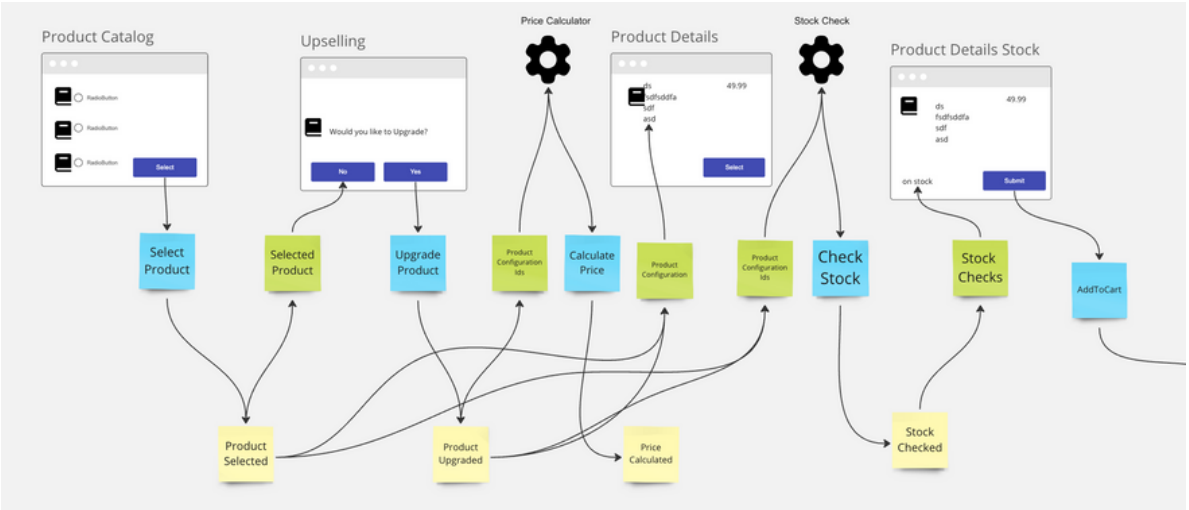


Commands werden mit dem entsprechenden Event verbunden.

Idealerweise erkennt man direkt aus dem Modell, welches Element im Screen das Command gesendet hat.

SCHRITT 5 – WELCHE DATEN?

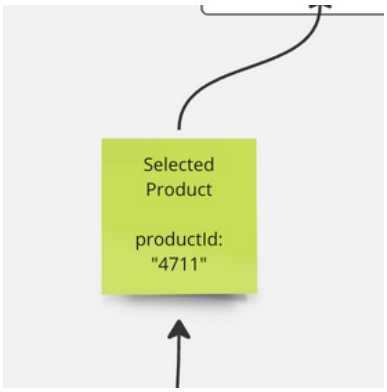




Was?

Dieser Schritt definiert, wie die Daten (“Read Model”) an den nächsten Screen bzw. den nächsten Slice weitergegeben werden.

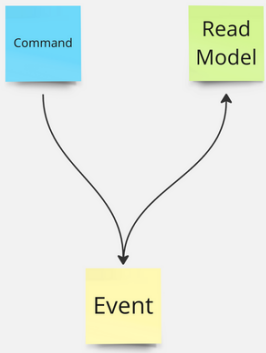
Die Read Modelle werden direkt mit den Events verbunden, aus denen sich die Daten zusammensetzen.

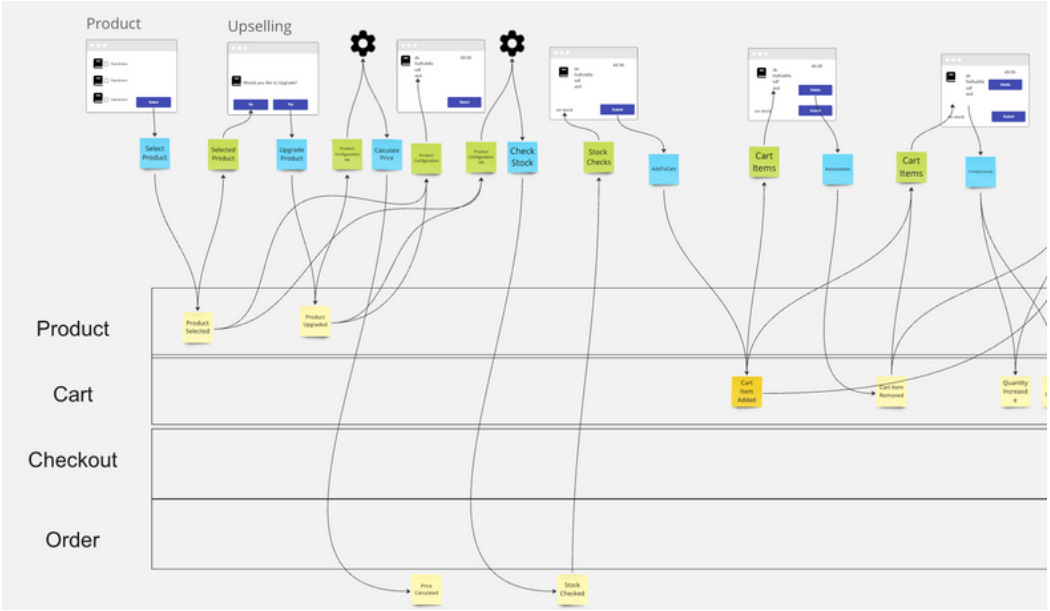


Hier gilt erneut, je detaillierter desto besser. Beim Befüllen der Read Modelle mit Testdaten fallen fehlende Schritte in der Analyse schneller auf.

“Die Read Modelle befüllen sich allein aus den Events der vorigen Schritte”

SCHRITT 6 – WELCHE SYSTEME?





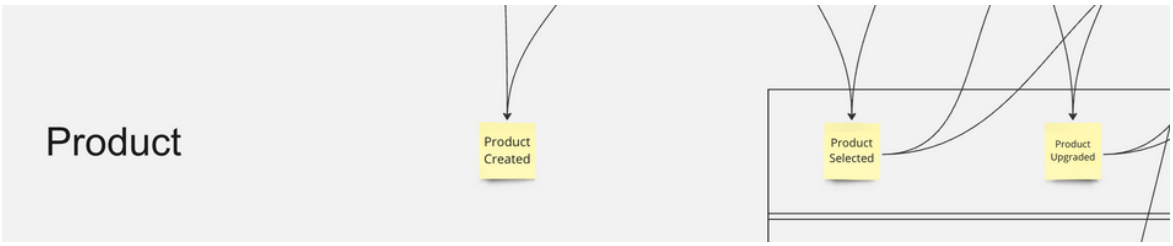
Was?

In diesem Schritt definiert ihr die “Swimlanes”

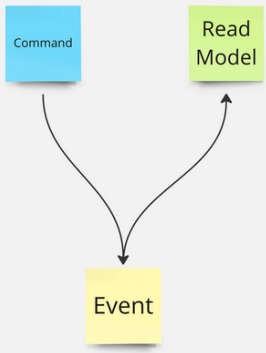
Swimlanes sind beteiligte Systeme bzw. Kontexte.
Oft sind Swimlanes eigene Module oder sogar Microservices, meist aber mindestens eigene Eventstreams.

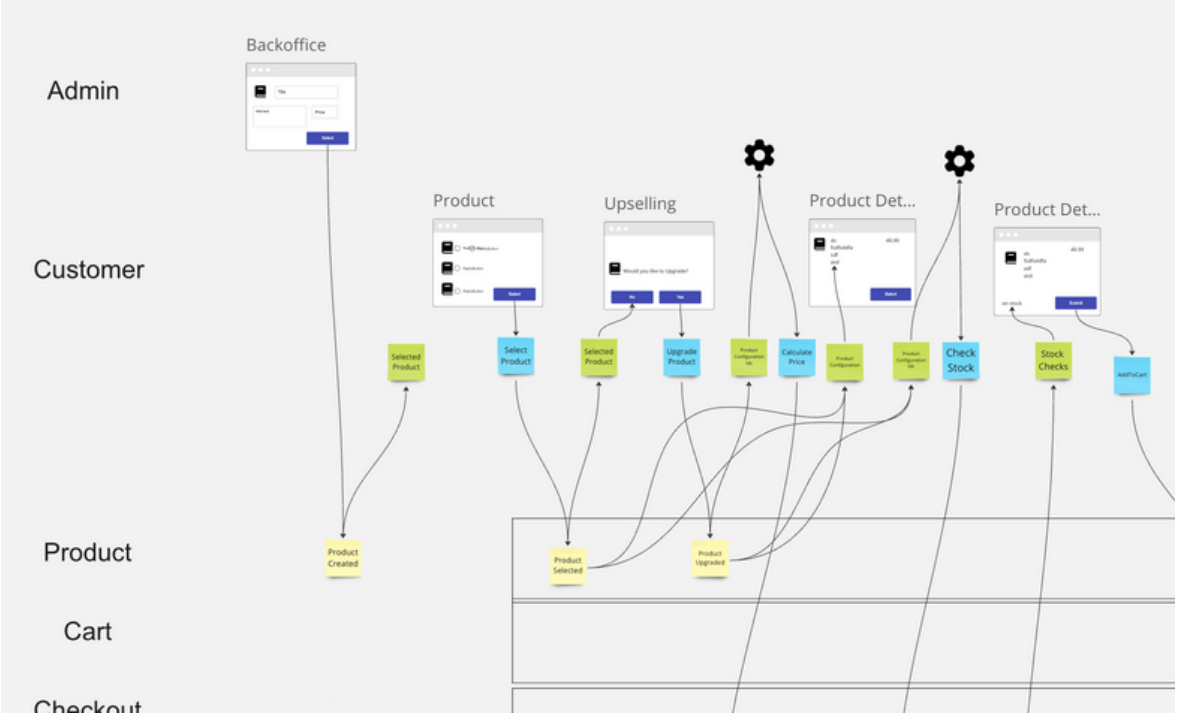
Swimlanes werden unterhalb des Diagrams definiert. Jedes Event ist eindeutig einer Swimlane zugeordnet.

Im Beispiel sehen wir “Product Created”, “Product Upgraded” die beispielsweise der Swimlane “Product” zugeordnet sind.



SCHRITT 7 –
WELCHE TEILNEHMER?





Was?

Oberhalb des Eventmodells definieren wir die Teilnehmer im System.

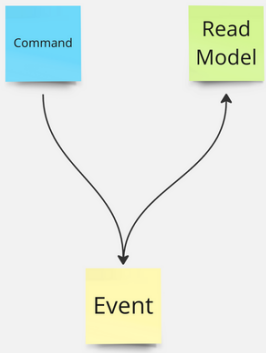
Üblicherweise haben wir eine Zuordnung von Screen zu einem bestimmten Teilnehmer.

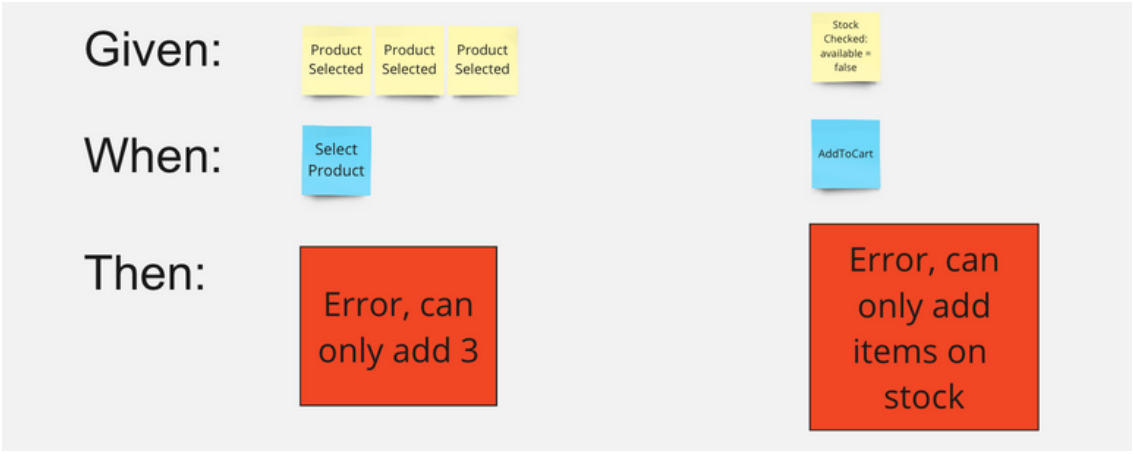
Im Beispiel hier sehen wir die Zuordnung zu Customer (Customer-Facing) und Admin (Backoffice).

Unterschiedliche Teilnehmer interagieren typischerweise mit unterschiedlichen Systembereichen.

Durch die Definition von Teilnehmern und der zugeordneten Events und Swimlanes ergibt sich ein Gesamtbild wie Daten durch das System fließen.

SCHRITT 8 – WELCHE POLICIES?





Was?

Im letzten Schritt definieren wir die Regeln (“Policies”) im System.

Policies werden wenn möglich unterhalb der Swimlanes definiert.

Im Beispiel beschreiben wir die Regeln mit “Given - When - Then”.

“Given”:

Wir haben drei Events vom Typ “Product Selected”

“When”

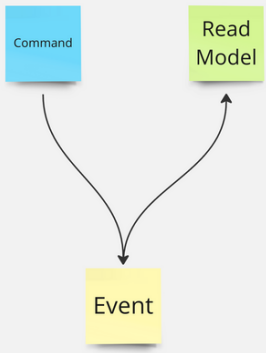
Angenommen wir schicken ein “Select Product” Command ins System.

“Then”

In diesem Fall erwarten wir einen Fehler, da nur maximal 3 Produkte im Warenkorb liegen können.

Typischerweise wird für jede dieser Businessregeln im Anschluss ein eigener Test implementiert.

EVENT MODELING



Wo hilft das Eventmodell?

Backlog Validierung
Sind alle Anforderungen erfasst?

Dokumentation
Alle Anforderungen sind einfach und verständlich in einem Dokument.

Onboarding
Neue Mitarbeiter finden sich im Eventmodell sofort zurecht und lernen das System praktisch kennen.

Projektplanung
Auf Basis der umgesetzten Slices im Eventmodell lässt sich der Projektfortschritt ermitteln.

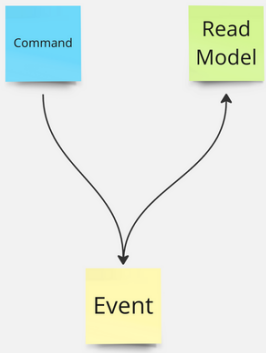
Implementierung
Das Eventmodell dient als Grundlage für die Implementierung.

Geschwindigkeit
Umsetzbare Anforderungen schon nach einem Tag möglich.

Qualitätssicherung
Entwickler und Tester wissen zu jeder Zeit welche zu testenden Effekte eine Aktion im System verursacht.

Skalierbare Entwicklung
In einer eventbasierten Architektur können Slices unabhängig voneinander entwickelt werden. Skalierung wird möglich.

LETZTER SCHRITT



Wer?

Ihr arbeitet in Java Teams und seid mutig genug, auch mal neue Dinge auszuprobieren um **alte Probleme zu lösen?**

Ihr wollt eine bessere Planung eurer Softwareprojekte ermöglichen?

Ihr wollt zuverlässige Zahlen gegenüber dem Management kommunizieren können?

Ihr habt den Mut auch ohne stundenlange Meetings produktiv zu arbeiten?

Ihr seid offen für moderne Konzepte, wenn sie euch wirklich helfen?

Was?

Willst du mehr über diese Art arbeiten erfahren wird es Zeit, dass wir uns unterhalten.

Buche mit dem unteren Link einen kostenlosen Info Call und ich gebe dir weitere Einblicke wie die Arbeit mit Eventmodeling bei dir in der Softwareentwicklung aussehen kann.

Martin Dilger

https://bit.ly/nebulit

https://eventmodeling.org

NEBULIT

Nebulit GmbH
Jahnstraße 4
83119 Obing
info@nebulit.de
www.nebulit.de