

**ITI8740/ITX8522: Software Development Team Project. 2026**

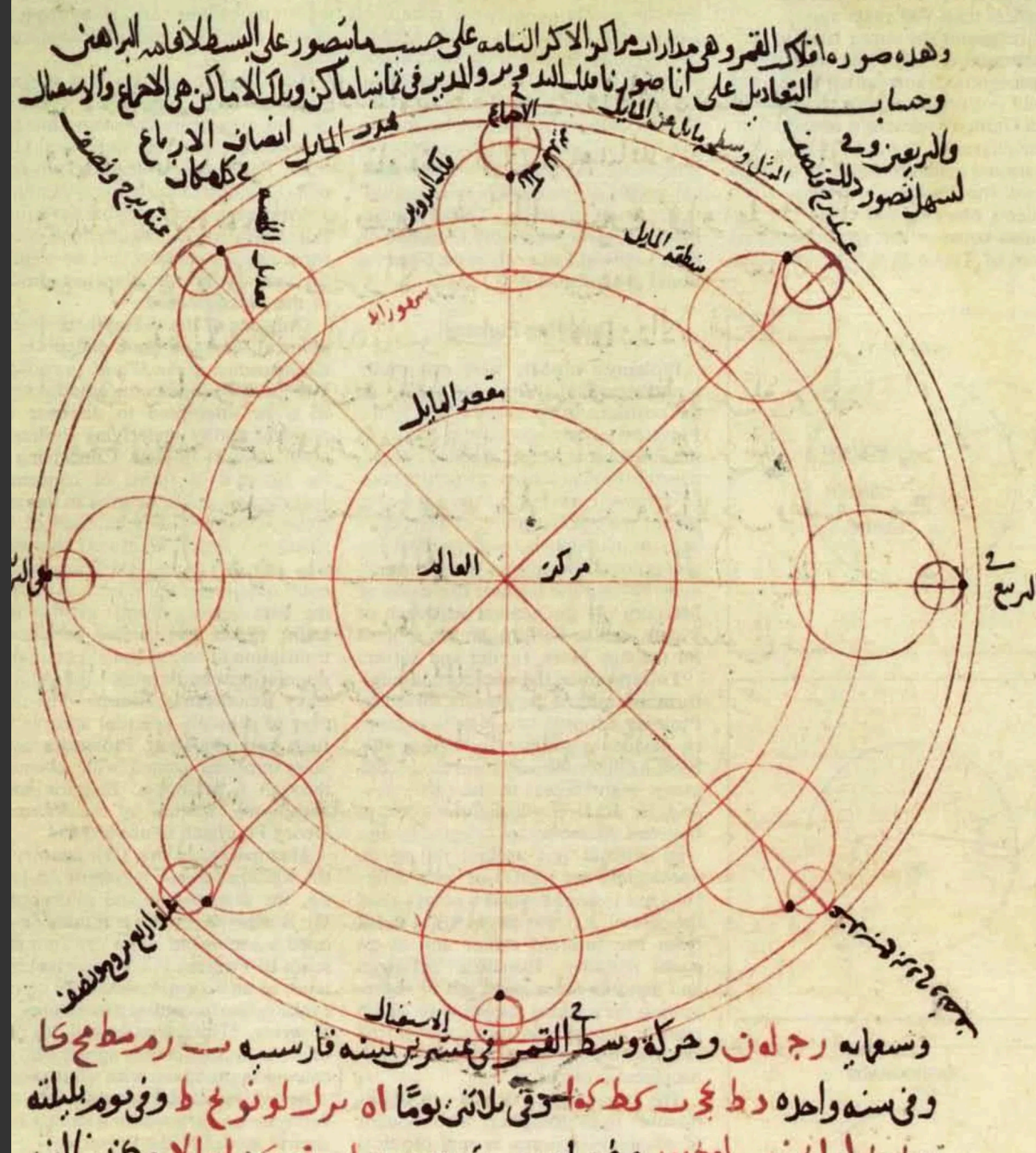
# **07. Modelling**

Martin Verrev

[martin.verrev@taltech.ee](mailto:martin.verrev@taltech.ee)

**The most important role of deliverables is to document your product. They help to communicate key concepts and the project's direction. They reduce product risk by providing a common understanding and recording decisions made throughout the process.**

**Modelling** is the act of constructing and using models — conceptual, mathematical, physical, or computational — to describe how something works or behaves under certain conditions.



# Model

A model is an **abstract** view of a system that ignores system details. Complementary system models can be developed to show the system's context, interactions, structure and behavior.

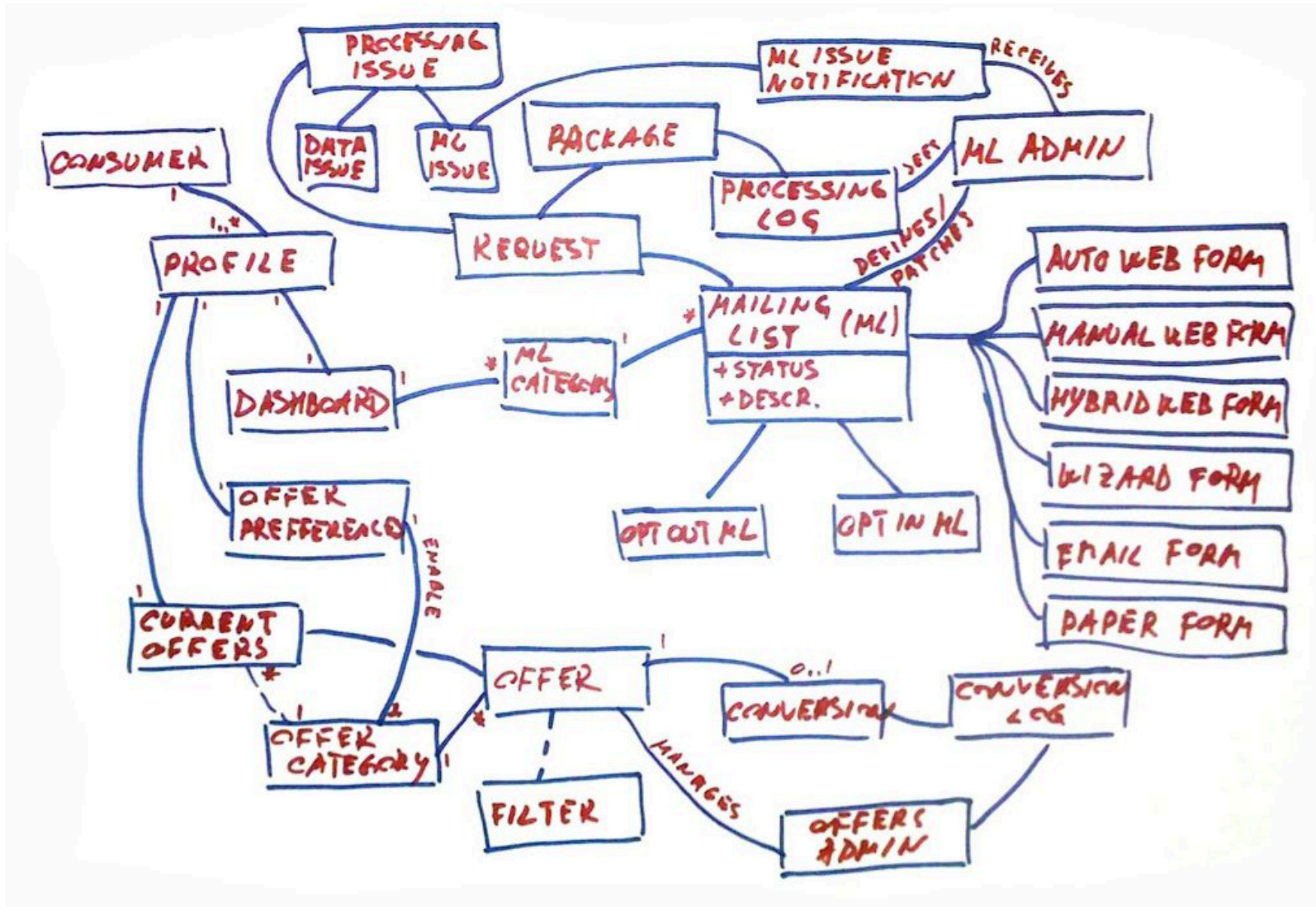
# Bird's Eye View: Domain Model

**Domain Modeling** is a way to describe and model real world entities and the relationships between them, which collectively describe the problem domain space.

**Domain** is a collection of related concepts, relationships, and workflows.

See also: <https://olegchursin.medium.com/a-brief-introduction-to-domain-modeling-862a30b38353>

# Example: Informal Domain Model



# Domain Model

Similar to class diagram or ERD, but includes also actors and focuses only on high level details. Domain model must be understandable to all stakeholders involved in the project.

Elements:

- **Concepts/Entities** are represented as classes
- **System roles** represented as actors
- All **connectors** are labelled and directed and start from actors

# Domain Model: Rules

- Classes denote concepts
- Class methods (used sparingly) denote important activities
- Class attributes (used sparingly) denote important values
- Connectors are:
  - Always directed
  - Always labelled
  - May denote multiplicity if it is relevant to the system
- All concepts must be connected to real world objects

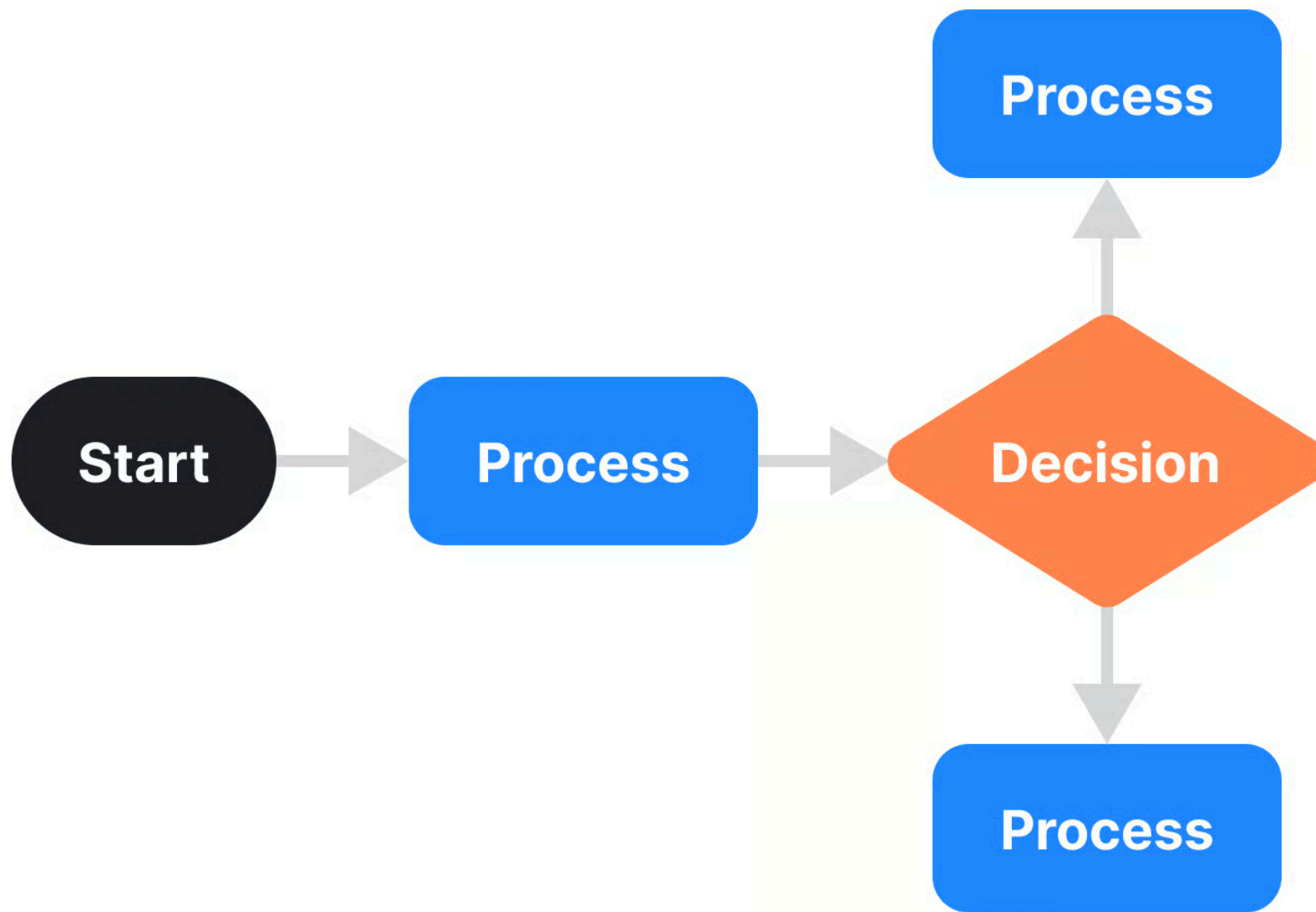
**Let's see: Domain Model Examples**

# Modelling Tools

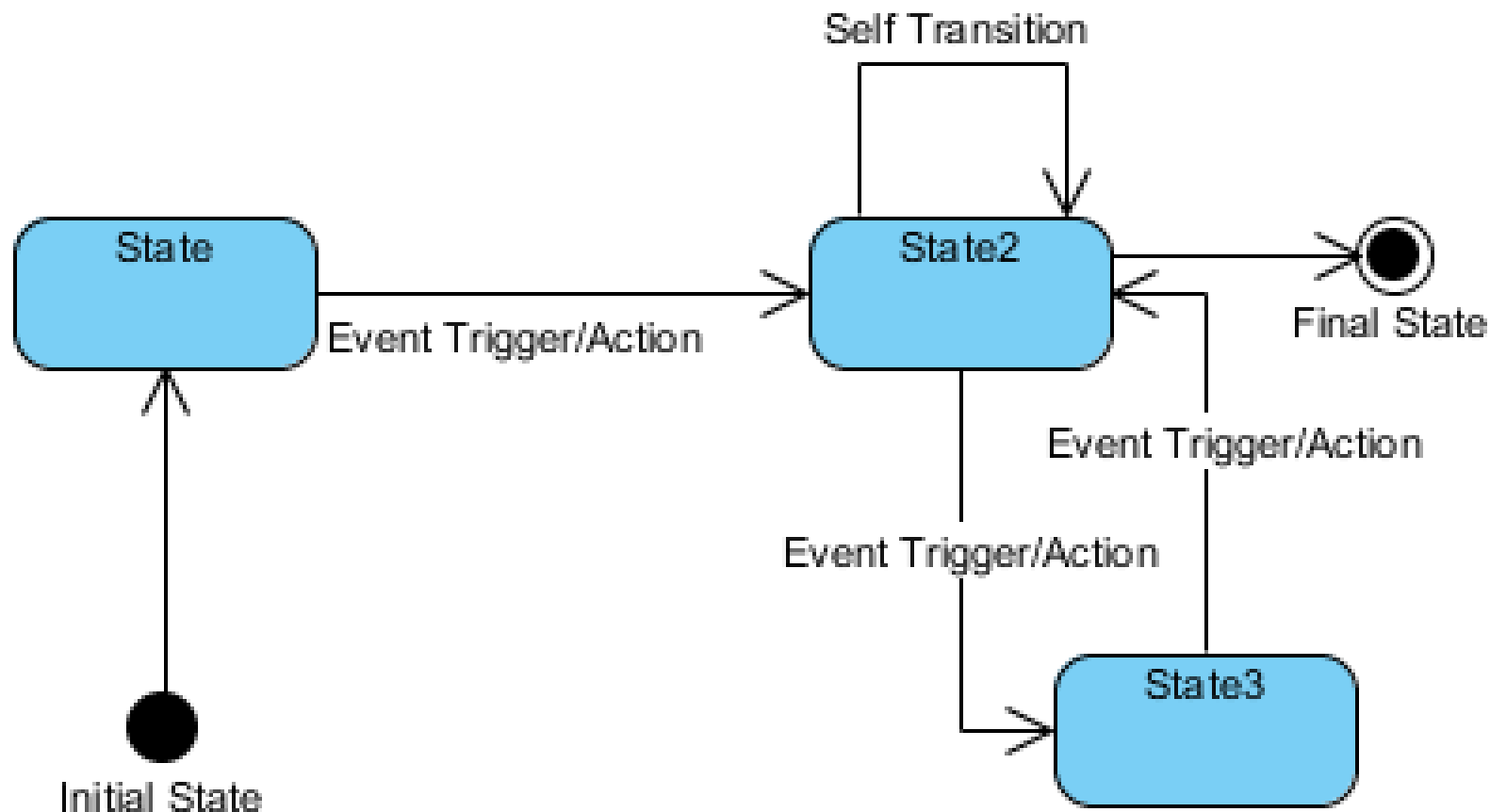
- [Enterprise Architect](#): Windows only, cumbersome, paid
- [Lucid](#): web-based. Paid, but free edu plan available.
- [StarUML](#): all platforms, paid
- [Draw.io](#): web based, free
- [Visual Paradigm Community](#): all platforms, free community edition available
- [Nomnoml](#): Open source (nodejs), lightweight, customizable. VS Code plugin available.
- [Mermaid](#): Open source (nodejs), powerful, lightweight, customizable. Numerous IDE plugins and LLM integration available. See also: <https://mike-vincent.medium.com/llm-mermaid-how-modern-teams-create-uml-diagrams-without-lucidchart-e54c56350804>

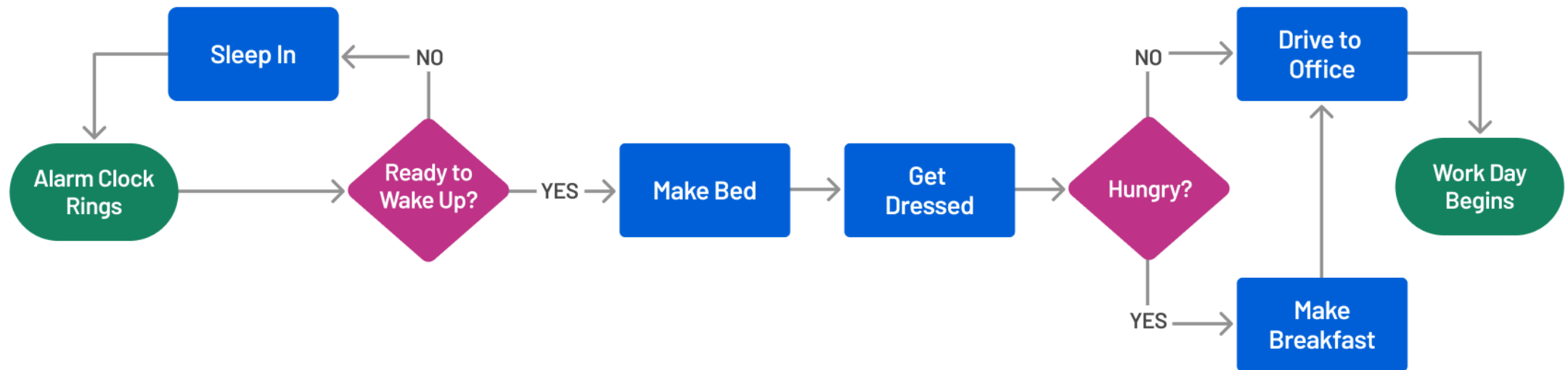
## Modelling User Actions

**User flow** (*kasutajavoog*) is diagram showing a set of interactions that describe the typical or ideal set of steps needed to accomplish a common task performed with a product. The focus is narrowed to a specific objective within one product.



**User flow is similar to a simplified UML state diagram.**





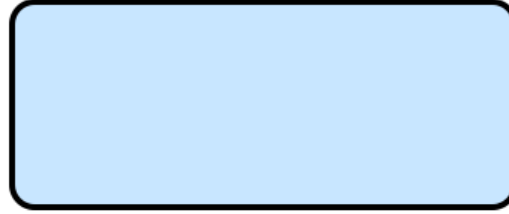
# Key Elements of a User Flow

Whenever you visualize a user flow, these are the must-have components.

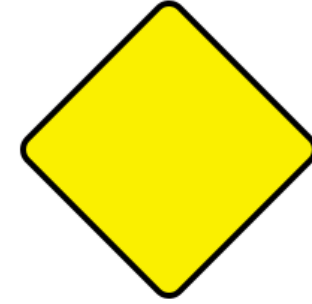
- **Entry Points:** Where the user flow begins, whether users are landing on your homepage, clicking an email link, or opening your app.
- **Decision Points:** are the critical points where users need to make choices, such as signing up or selecting between different paths.
- **Actions and Interactions:** These are the specific tasks users must complete, like creating an account, making a purchase, or setting up their profile.
- **Arrows** are the connectors that guide users from one step to another. and show the flow of actions and decisions showing how each part of the journey is linked.
- **End Goal:** The ultimate objective you are aiming to achieve with your app.



START/END



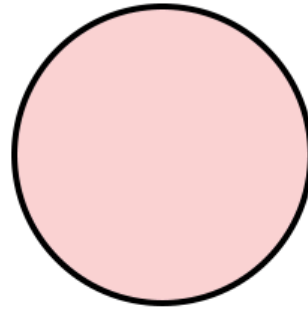
PROCESS / ACTION



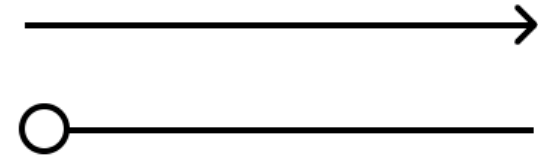
DECISIONS



INPUT / OUTPUT



CONNECTOR



CONNECTIONS / ARROWS

# How to make a user flow diagram

When crafting your user flow diagram, discuss the following with your team:

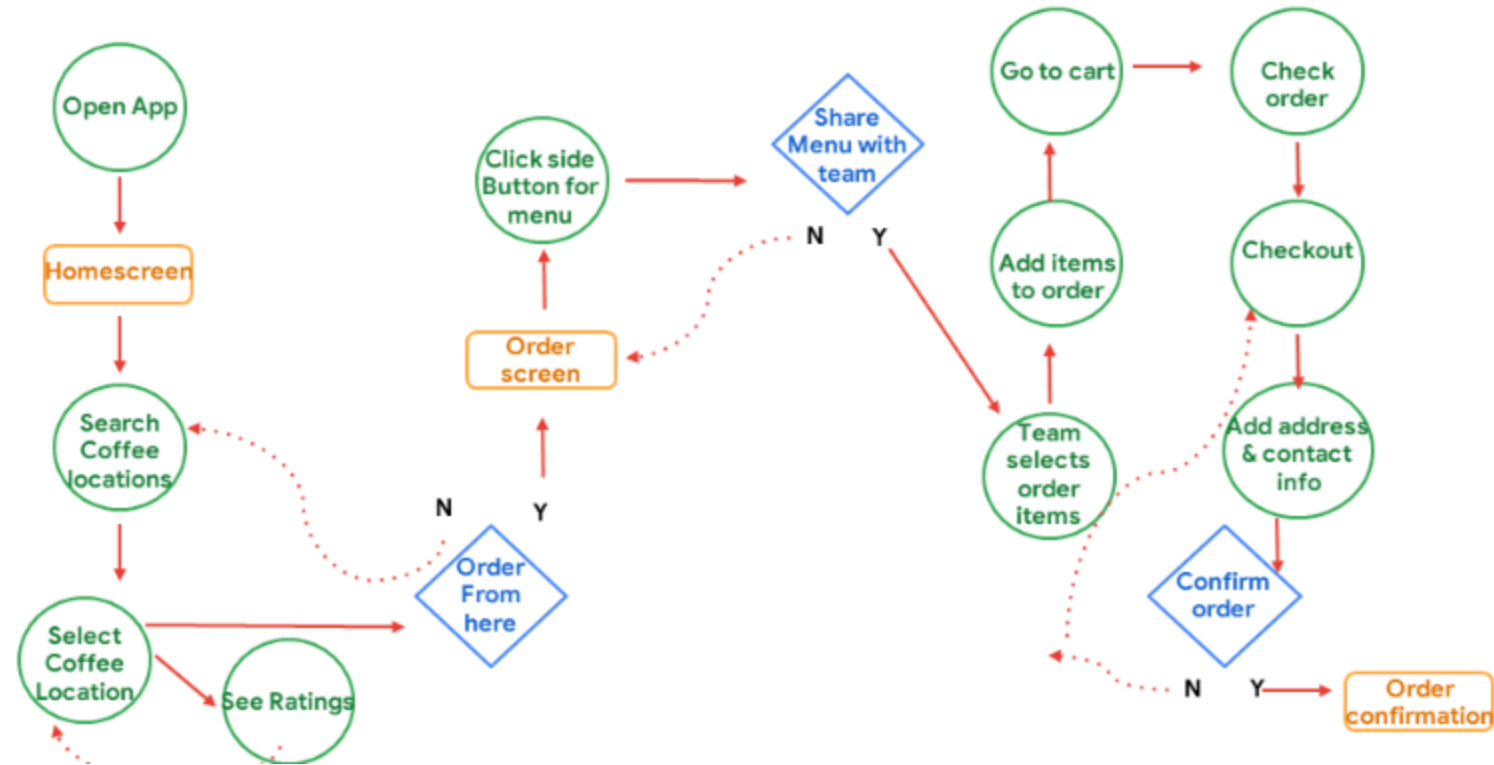
1. **What is the user trying to do?** — Determine your objective and your users' objectives.
2. **What is the entry point into this flow?** — Identify where your users are coming from.
3. **What extra information will the user need to be successful?** — Identify what information your users need and when they need it.
4. **What are the user's hesitations or barriers to accomplishing the task?** — Identify the challenges and keep them in mind when designing the full experience.

# Designing Flows - Considerations

1. Descriptive name.
2. One directional flows.
3. One goal per flow.
4. Always have a legend key for users who are not familiar with such deliverables.
5. Decide and entry point
6. Clearly label each action
7. Use colors wisely.
8. Always make sure your user flows are complete when it comes to a goal.

## User flow

**User task:** Use the Coffee House app to quickly and easily make multiple orders in an organised way.



<https://matomo.org/blog/2024/10/understanding-user-flow-eight-practical-examples-for-better-ux/>

# Example: Zoom Video Call Flow

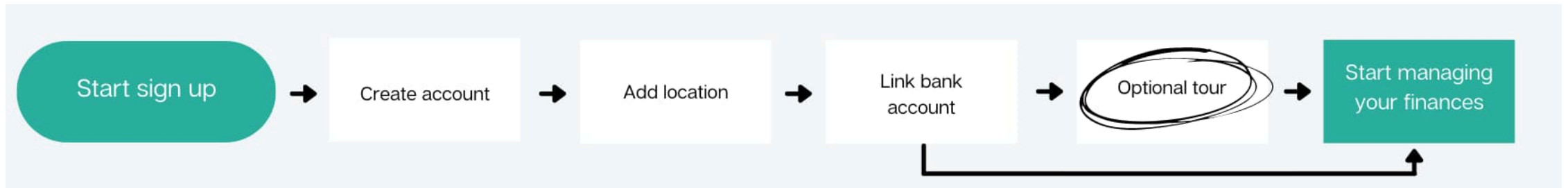
Zoom is easy to use. In five clicks or less, you can open a room and invite all your friends and colleagues to a video chat.



There's no barrier between what the user wants (to start a call) and what they need to do. No windows ask whether you want to invite person x or disable option y—just start the meeting and go.

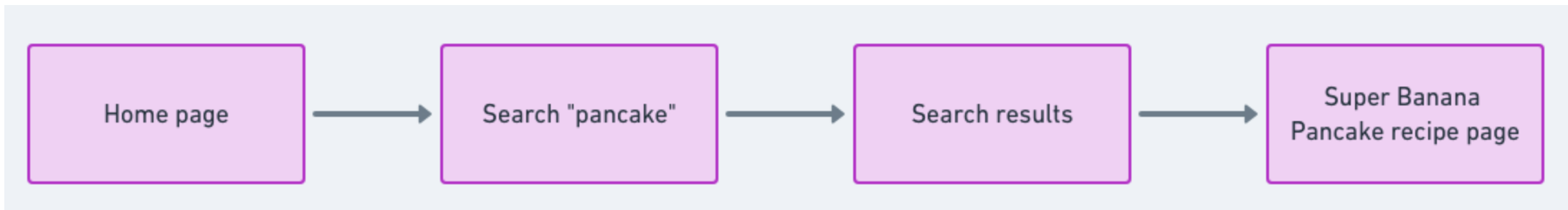
# Example: Mint's product tour

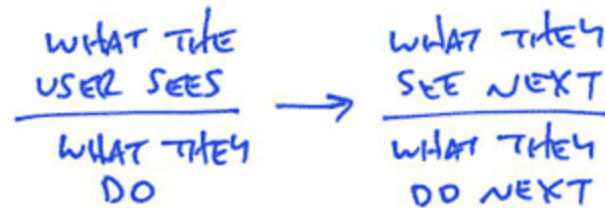
Mint has designed an optional product tour that shows off its features while educating its audience.



The optional tour can feel a bit extra when it comes to a user onboarding flow. It's the only part of the flow that's completely optional. However, Mint uses it as a life preserver for those financial newbies drowning in unfamiliar terms that'll determine their ability to buy a house or retire.

# Example: Finding Pancake Recipes

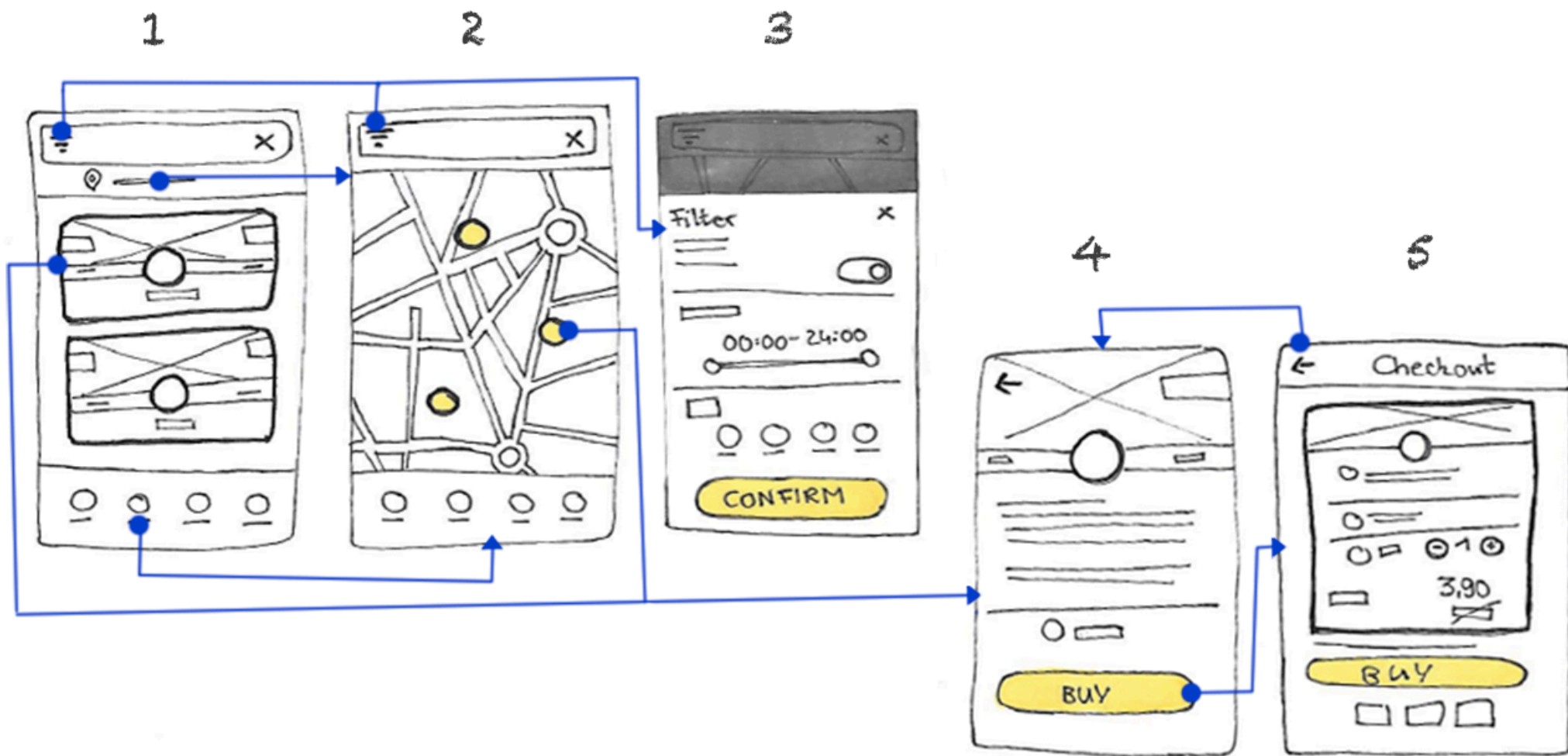




From: A shorthand for designing UI flows. <https://signalvnoise.com/posts/1926-a-shorthand-for-designing-ui-flows>

# Modelling Application Behavior: Wireframe

A wireframe is a low-detail representation of an interface. It omits color, image detail, and other visual design specifics, providing instead a simple inventory of what's on the page and how it should be laid out.



# Wireframe - Considerations

1. **Information organization.** Which items should be grouped and where? Are there any particular relationships that need to be made more evident than others? How should these groups be prioritized?
2. **Content.** What content needs to be present on the page? Will it be prose alone or does the page need to accommodate images and video?
3. **Functionality and controls.** What can users do on the page? How will users navigate the site? Is there a search function? A log-in control? Are there any inputs such as forms?

# Wireframe - Considerations

4. **States.** What are the various states of the page? How do forms handle errors?  
Does the page vary depending on the user's status - e.g. logged in versus logged out?
5. **Behavior.** Are there interactions that happen without a full page refresh? How does the page respond to input?
6. **Metadata.** What page is this? How does it relate to the sitemap? What project does it belong to? Who is the author? What version is it?
7. **Annotation.** Nuanced interactions or complex points may need further explanation. Use annotation callouts to highlight these areas.

# Wireflow

**Wireframe + User Flow**

ST

login

password

(+)

user auth

Error

Connection  
Cannot be  
Established

(frowning face)

No connectivity, auto  
recovery

SETTINGS

FIRST NAME

MIDDLE NAME

LAST NAME

GENDER ☒ ☐

TERMS ☐ SIGN ☐

initial setup/  
TnC screen

Details to be  
asked TDC, but  
probably grouped

TERMS

|||||

|||||

|||||

|||||

Terms & Privacy  
(TEXT PAGE)

HISTORY EDIT

PROVIDE  
DETAILS

PROVIDE DETAILS

SELECT

☒ HOTEL ☐ OTHER

☐

☐

☐

User selects  
what details  
to provide to  
service provider

SUBMITTING

Your Code

2A316B

CANCEL

CODE DISPLAYED  
IS USED BY  
SERVICE PROVIDER

MY HISTORY

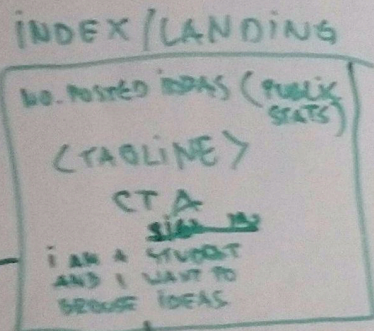
TALLINE 13.10.17 OR

OLYMPIA 10.10.17 OR

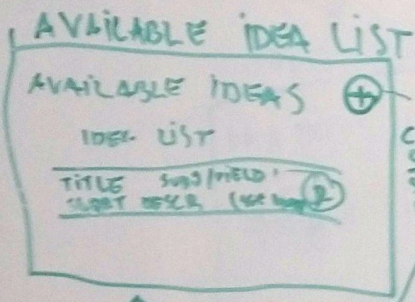
WILLOW INN 12.07.17 Pending

HISTORY. HIGHLIGHT  
WHERE CC DETAILS  
PROVIDED

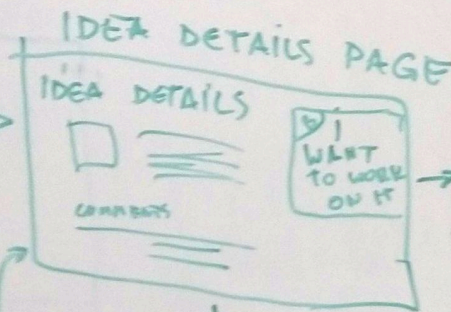
NAVIGATE ON success



GET: /ideas

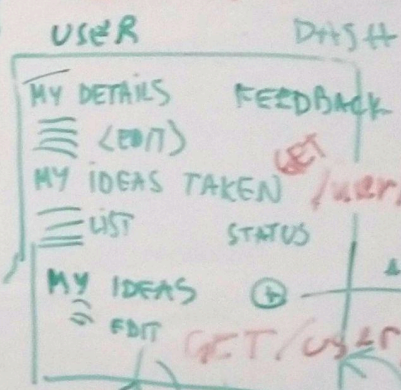


GET /ideas/:id



if user is not logged in, then redirect to login

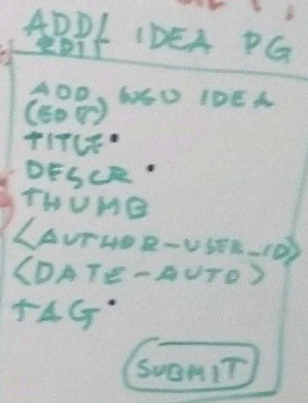
GET /user/:id



(+) - ADD IDEA

GET /idea/?id

GET /tags



if ID is then else

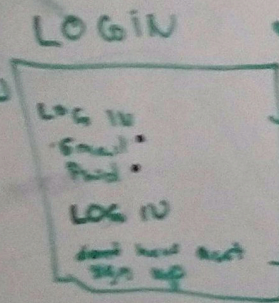
GET:

POST:

PUT:

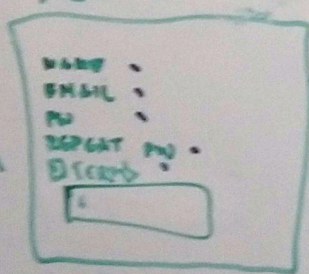
: param

? param



BACK

SIGN UP

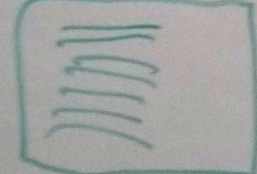


USER WANTS TO ADD IDEA

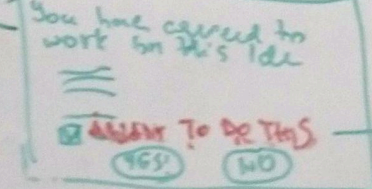
BACK ON SUCCESS

PUT /user

(MODAL)



CONFIRM (MODAL)



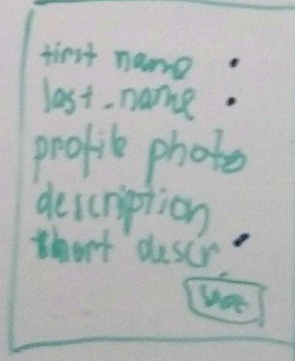
PUT /working

ACTION: NOTIFY

AUTHOR

GET /user/:id

EDIT PROFILE



YES

EDIT PROFILE

POST /user

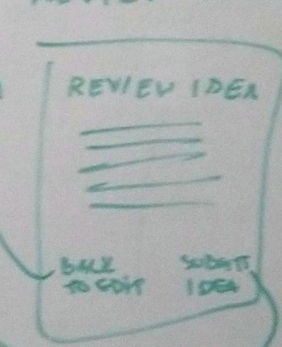
SUBMIT

PUT /idea/:id

POST /idea/:id

EDIT

REVIEW IDEA



USER - USER\_IDEA - IDEA

idea\_id

user\_id

status

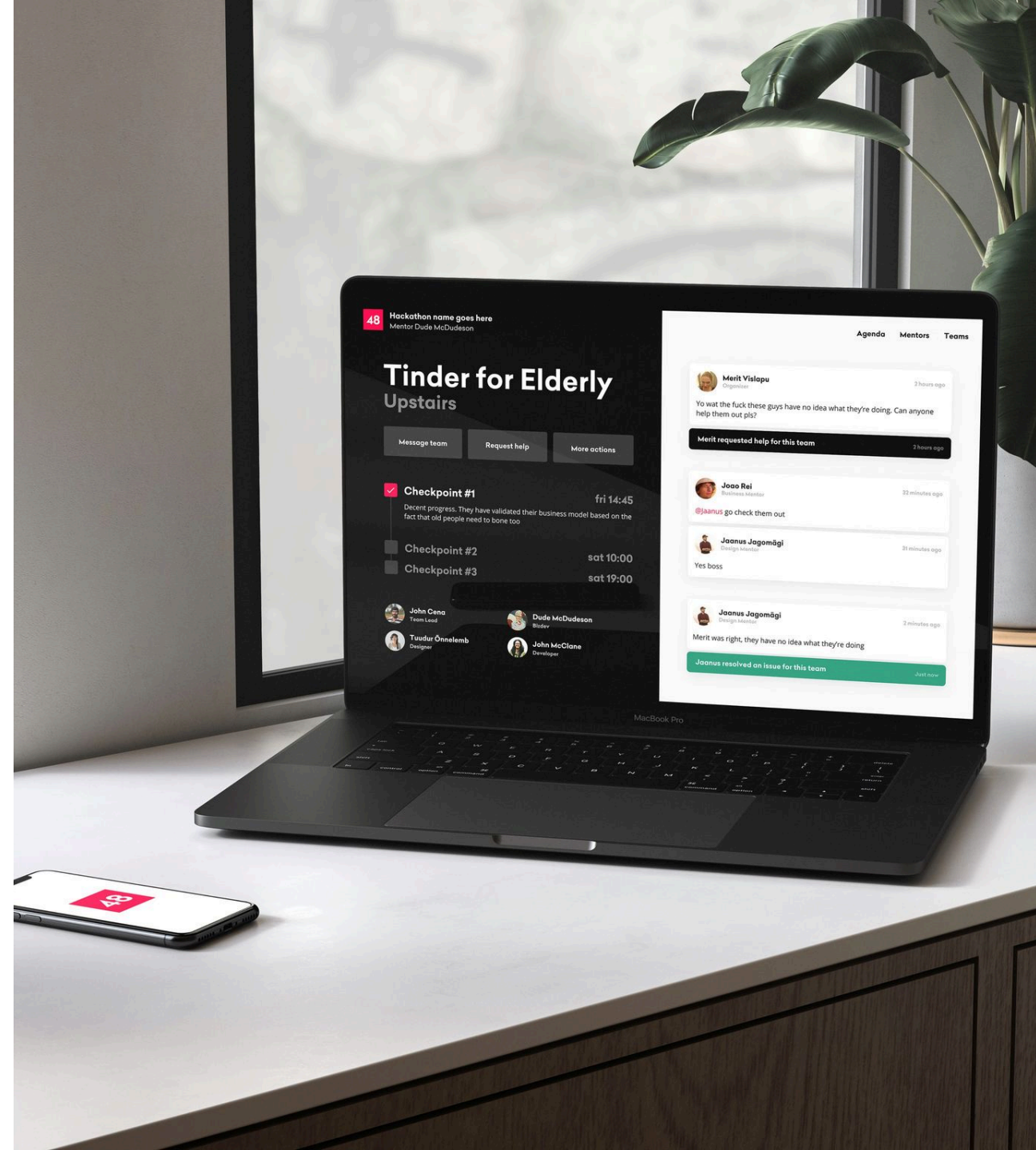
# Mockup

A mockup is the next, more in-depth iteration of the wireframe outline. A mockup is a static wireframe that includes more stylistic and visual UI details to present a realistic model of what the final page or application will look like.

# Mockup

A mockup typically includes additional visual details such as:

- Colors, styles, graphics, and typography
- Styled buttons and text
- Navigation graphics
- Component spacing







## Further Reading

- Domain Modeling: What you need to know before coding.  
<https://www.thoughtworks.com/en-th/insights/blog/agile-project-management/domain-modeling-what-you-need-to-know-before-coding>
- Domain Modeling by Example. <https://codeburst.io/rule-your-domain-model-d4beae6806c>
- A UX designer's guide to user flows. <https://uxdesign.cc/a-ux-designers-guide-to-user-flows-bbbc61f8b666>
- A shorthand for designing UI flows. <https://signalvnoise.com/posts/1926-a-shorthand-for-designing-ui-flows>

## Further Reading

- Sitemaps & Information Architecture (IA).  
<https://xd.adobe.com/ideas/process/information-architecture/sitemap-and-information-architecture/>
- Why the Best Wireframe Style Is No Style.  
<https://uxmovement.com/wireframes/why-the-best-wireframe-style-is-no-style/>

**Thank you!**