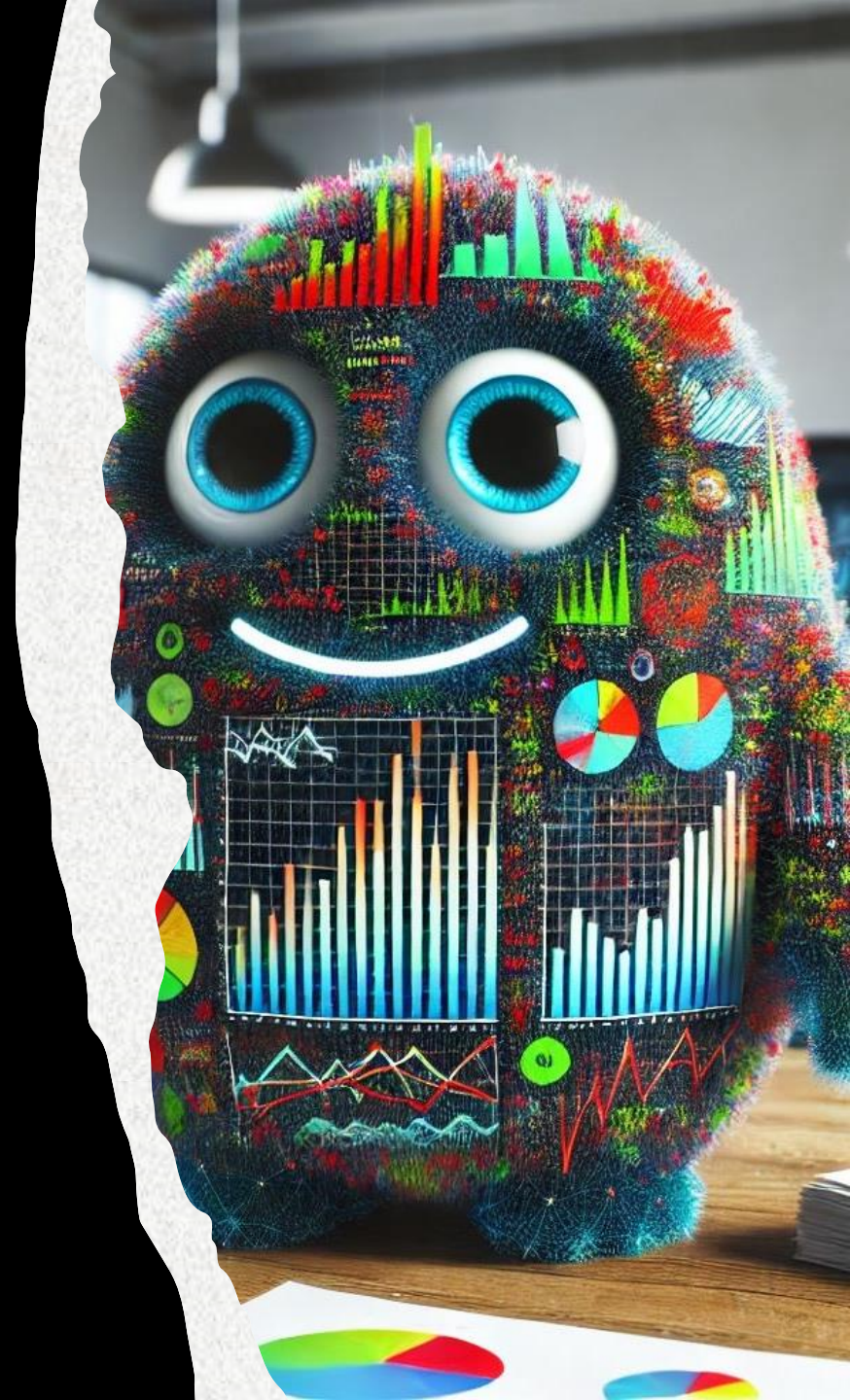


A Decentralized Love Story

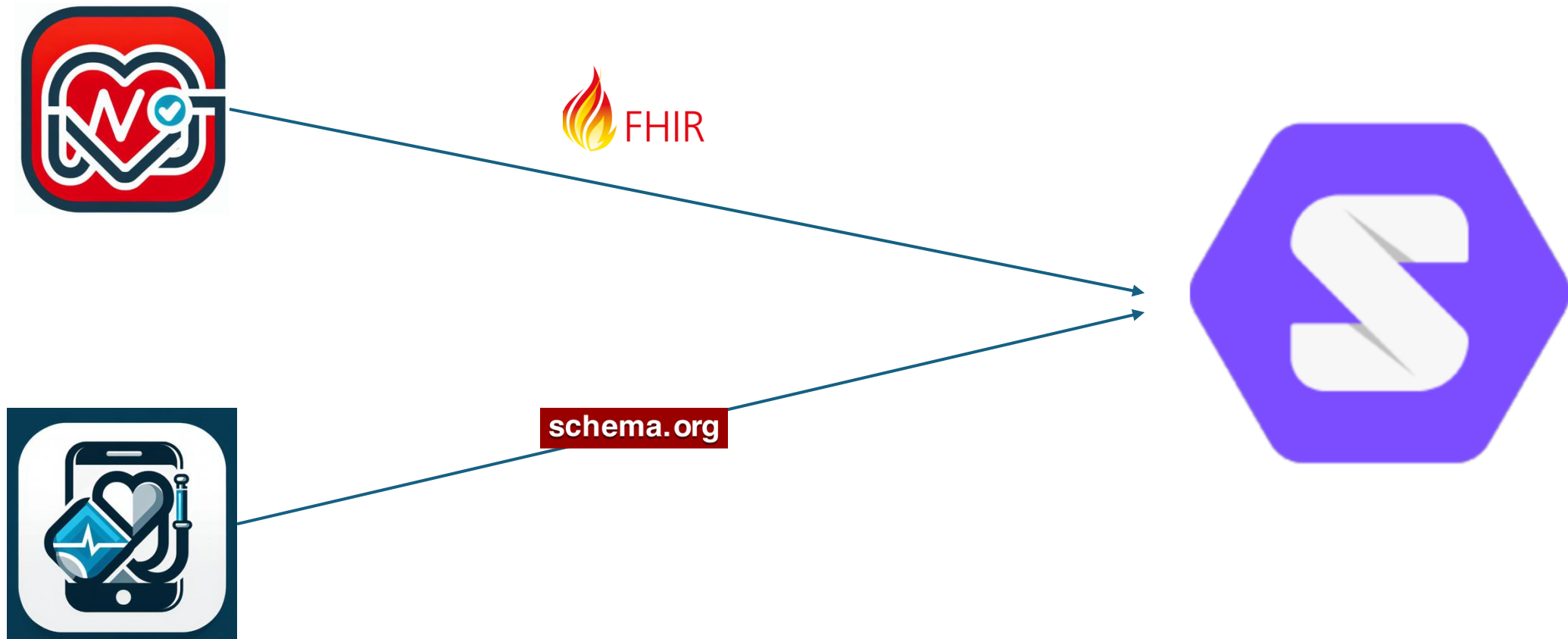


Who am I?



Chapter 1: What is your address?

Ontologies make data interoperable



Chapter 1: What is your address?



Address	Σ	N	Element
use	?!	Σ	0..1 code
type	Σ		0..1 code
text	Σ		0..1 string
line	Σ		0..* string
city	Σ		0..1 string
district	Σ		0..1 string
state	Σ		0..1 string
postalCode	Σ		0..1 string
country	Σ		0..1 string
period	Σ		0..1 Period

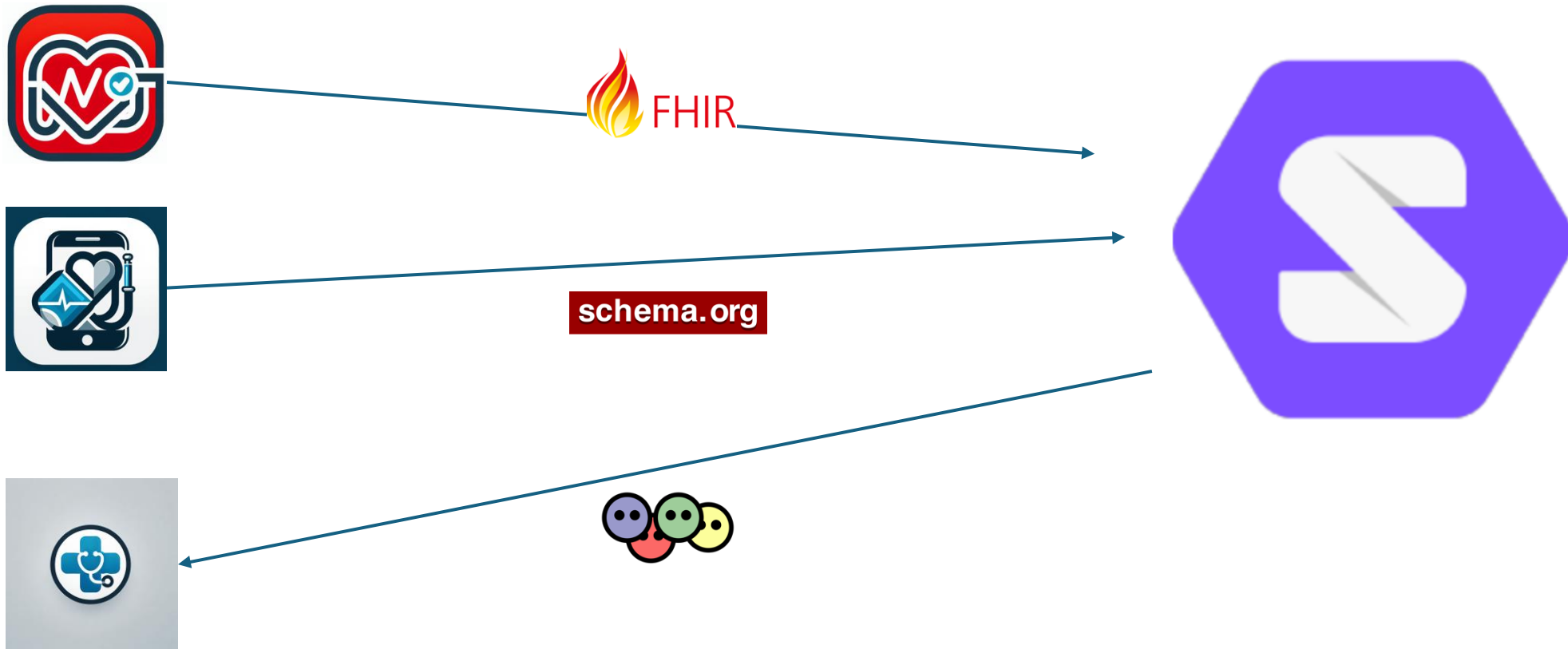


schema.org

Property	Expected Type
Properties from PostalAddress	
addressCountry	Country or Text
addressLocality	Text
addressRegion	Text
postOfficeBoxNumber	Text
postalCode	Text
streetAddress	Text

Chapter 1: What is your address?

Ontologies make data interoperable



Data feels
misunderstood

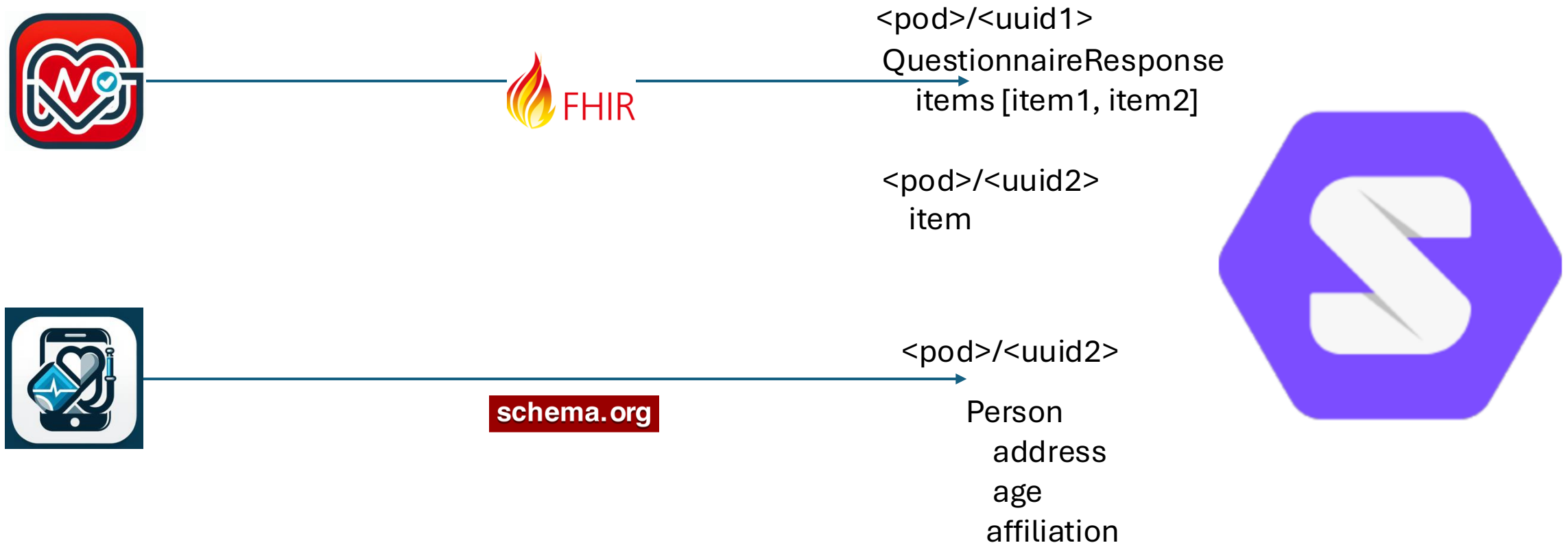




For your specific use case, there is an **optimal data model**. Data Loves business Logic.

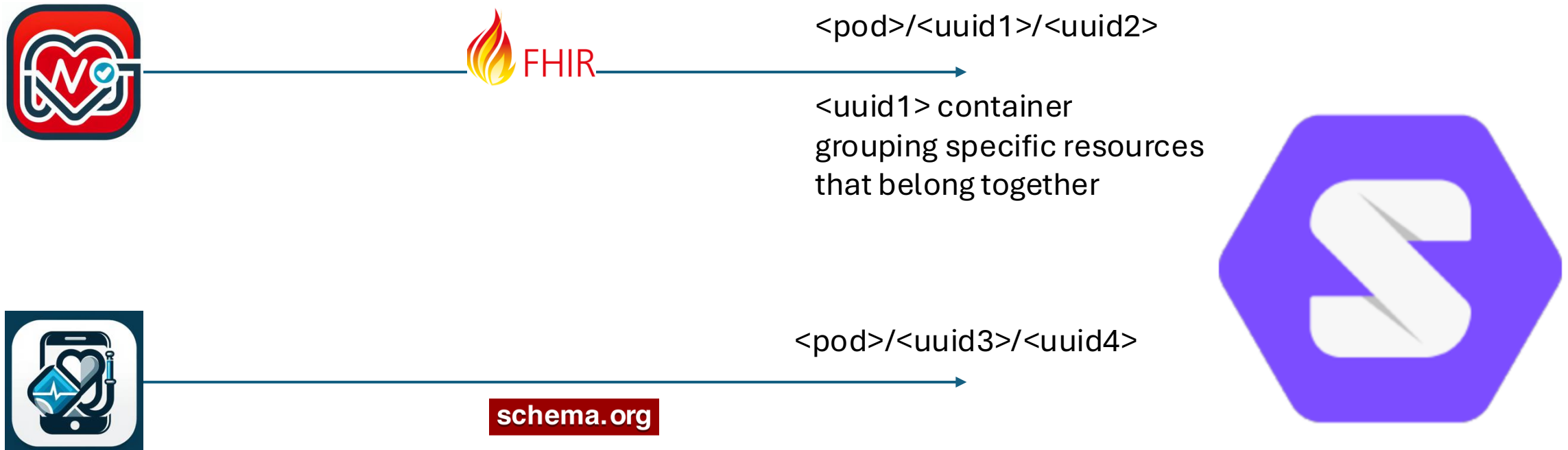
Chapter 2: Where to write in the pod?

Access to a container or resource



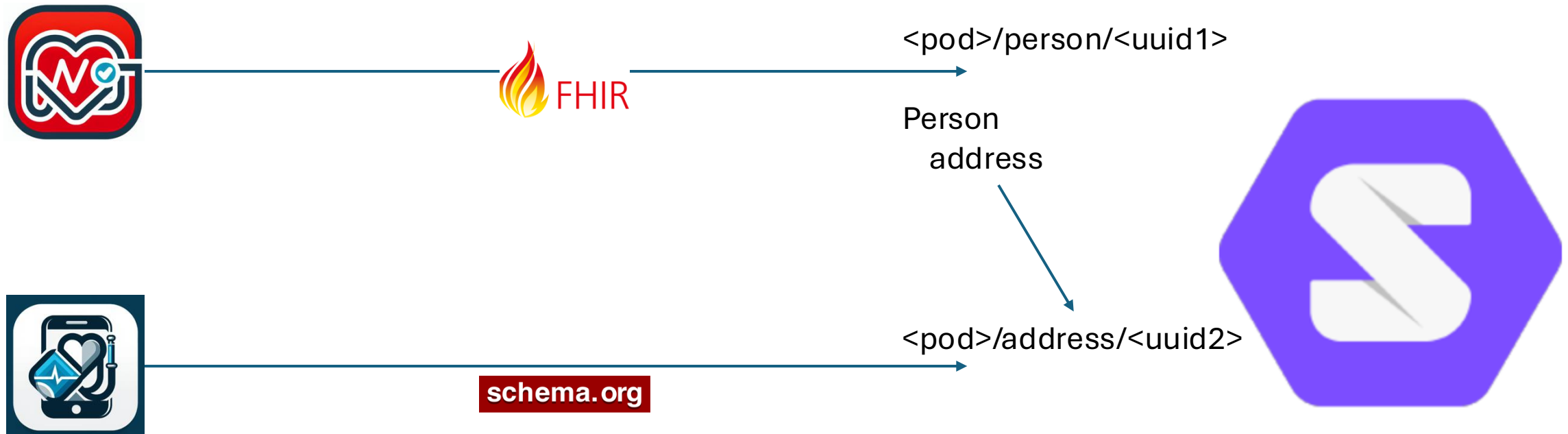
Chapter 2: Where to write in the pod?

Access to a container or resource



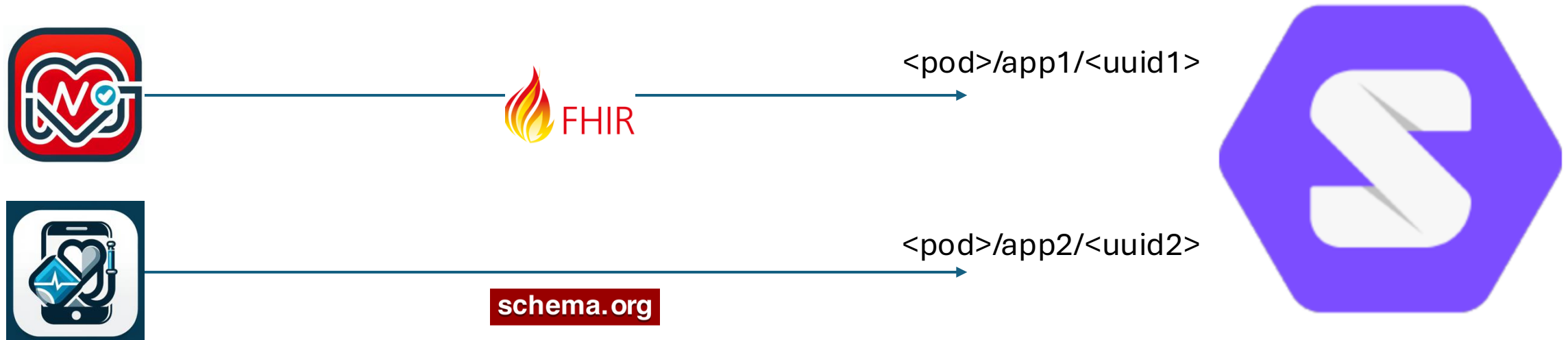
Chapter 2: Where to write in the pod?

Access to a container or resource



Chapter 2: Where to write in the pod?

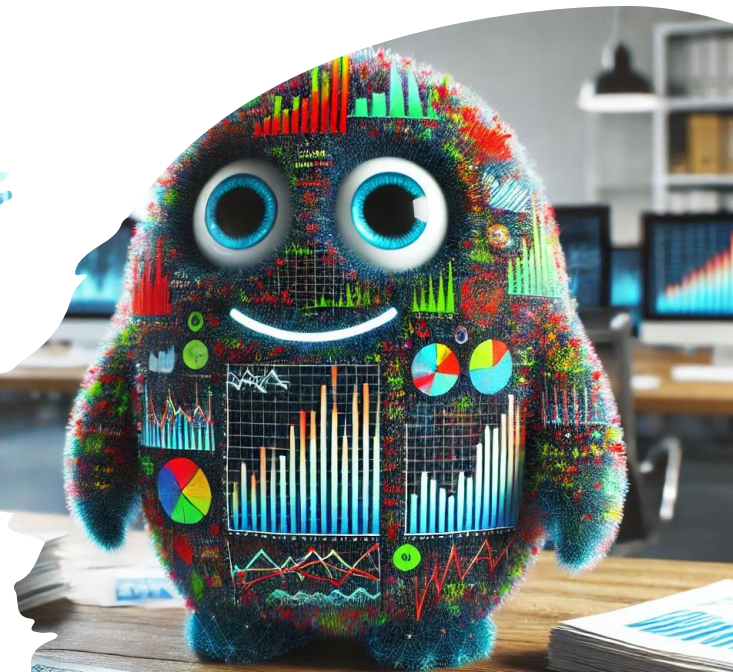
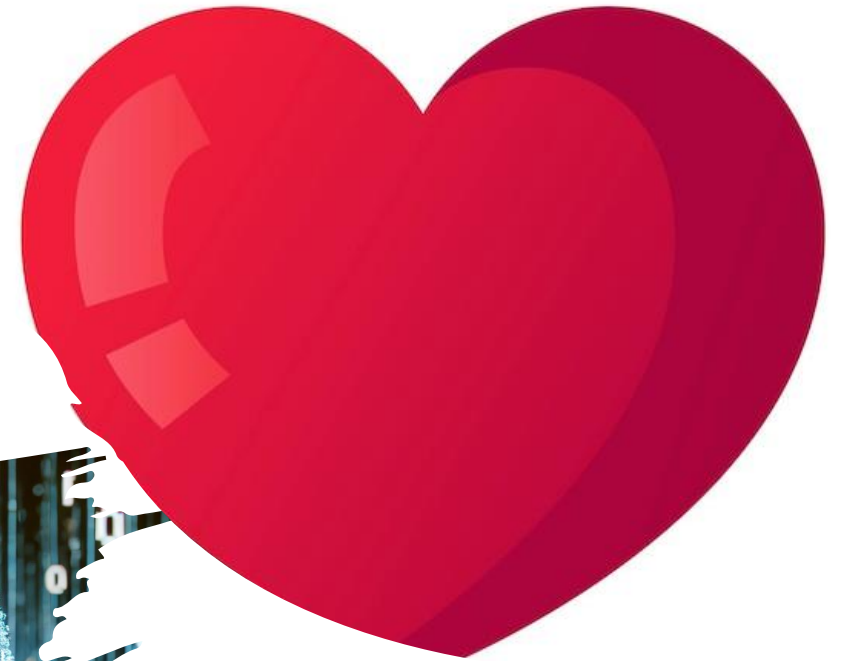
Access to a container or resource



Data feels lost
















The business logic allows you to **organize** your data. Without it, how to keep the data manageable?



Chapter 3: defining fine grained

Structure

Name	Flags	Card.	Type
 Patient			DomainResource
...  identifier	Σ	0..*	Identifier
...  active	?! Σ	0..1	boolean
...  name	Σ	0..*	HumanName
...  telecom	Σ	0..*	ContactPoint
...  gender	Σ	0..1	code
...  birthDate	Σ	0..1	date
...  deceased[x]	?! Σ	0..1	
...  deceasedBoolean			boolean
...  deceasedDateTime			dateTime
...  address	Σ	0..*	Address
...  maritalStatus		0..1	CodeableConcept

Person

A Schema.org Type

Thing > **Person**

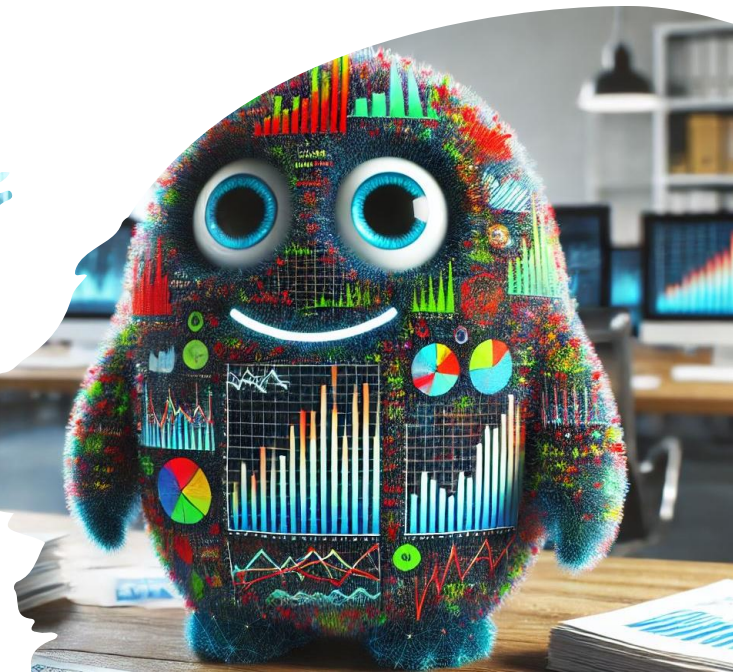
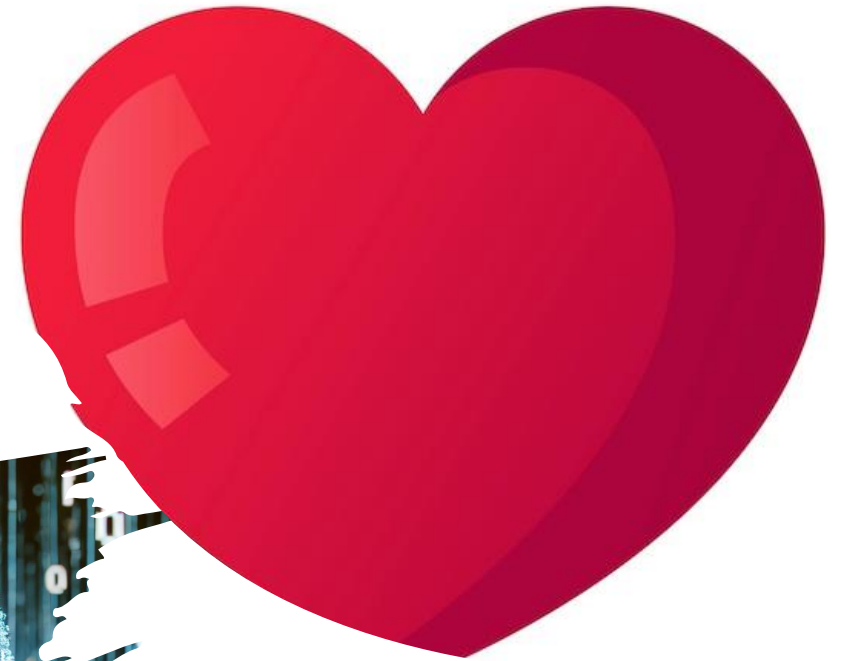
A person (alive, dead, undead, or fictional).

Property	Expected Type
Properties from Person	
additionalName	Text
address	PostalAddress or Text
affiliation	Organization
agentInteractionStatistic	InteractionCounter
alumniOf	EducationalOrganization or Organization
award	Text
birthDate	Date

Data feels
exposed



The business logic defines **what**
data can be shared for **which**
purpose and to **whom**.

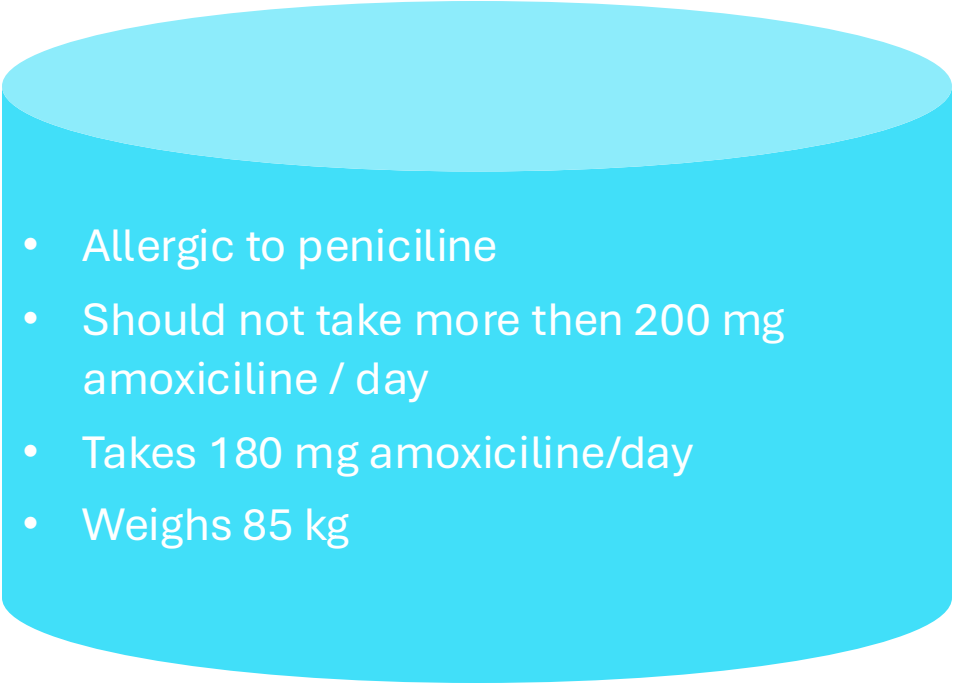


Chapter 4: Open ecosystem

- We Are defining strict rules and ontologies to make data interoperable and reusable and keep consistency.
- No technical protection whatsoever to guarantee nothing will happen from some other entry point...
- E.g.: medicine optimizer

Separating apps from the data

- New app **medicine calculator** who adjusts your dose based on your weight
- If the business logic of *user should not be prescribed...* is not enforced programmatically surrounding the data, it can lead to critical errors.
- If the business logic is encapsulated in the api and the data is accessible only via the api managed by the party with the business logic, this fault will never happen

- 
- Allergic to peniciline
 - Should not take more then 200 mg amoxiciline / day
 - Takes 180 mg amoxiciline/day
 - Weighs 85 kg

Data feels
abused



Data should be interpreted in a specific way. It is **guarded** by the business logic that accompanies the data.



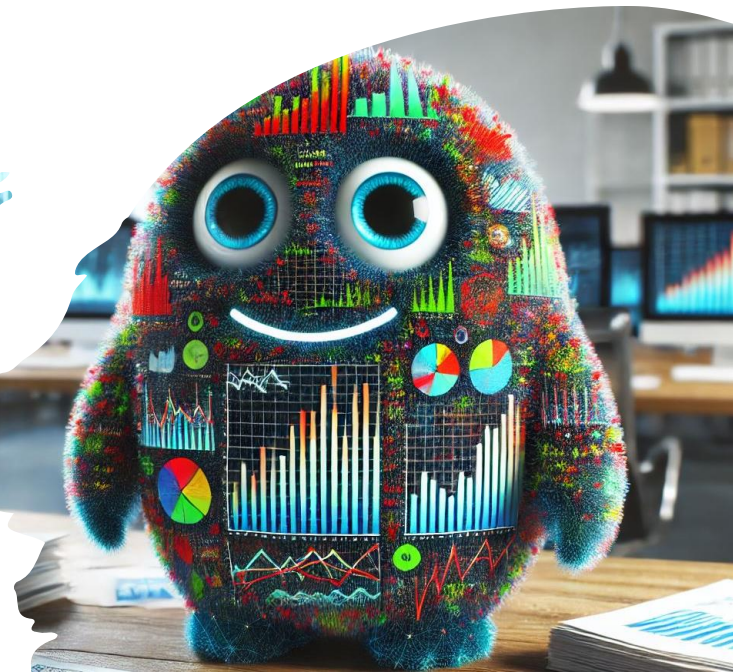
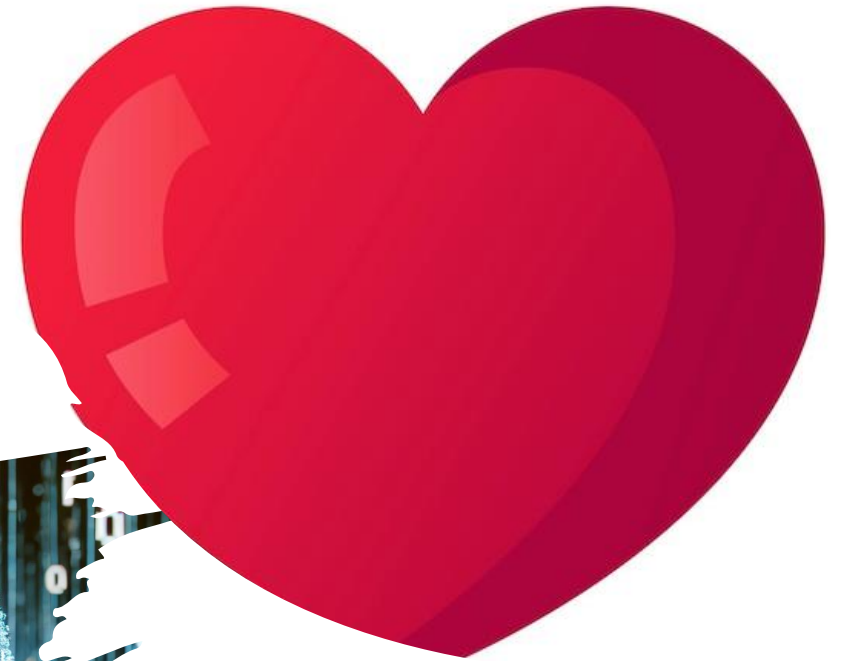
Chapter 5: sharing data for research

- No overview of who has a pod or what data.
- We can't send them a request to share the information
- Access to complete data pod to search for data

Data feels
violated



Data wants to be close to the business party that knows how to look for the right data to anonymize.



Chapter 6: What if something goes wrong

- There are persons with the task to fix data errors resulted by bugs in the programs or bugs in the procedures

Data is not to be
trusted anymore



The party with the business logic know-how can keep the data in a **consistent** state, overcoming bugs and error events.

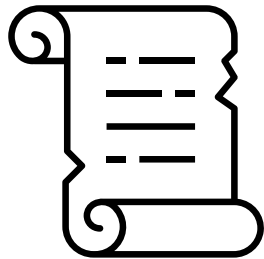


Conclusion

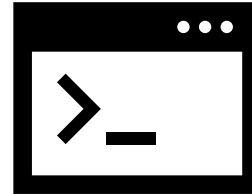
December 13, 2023

One of the key components of Solid is that it "separates the apps from the data storage", to quote Inrupt's co-founder Sir Tim Berners-Lee in a [Washington Post interview](#) a few years back.

Data



Apps



SOLUTIONS

Solution 1

It's not our problem



Solution 2

Webid,
Solid OIDC,
REST/url,
Access Requests,
Access Grants,
Verifiable Credentials



Solution 3

Keep digging for a solution to
encapsulating data with
business logic

