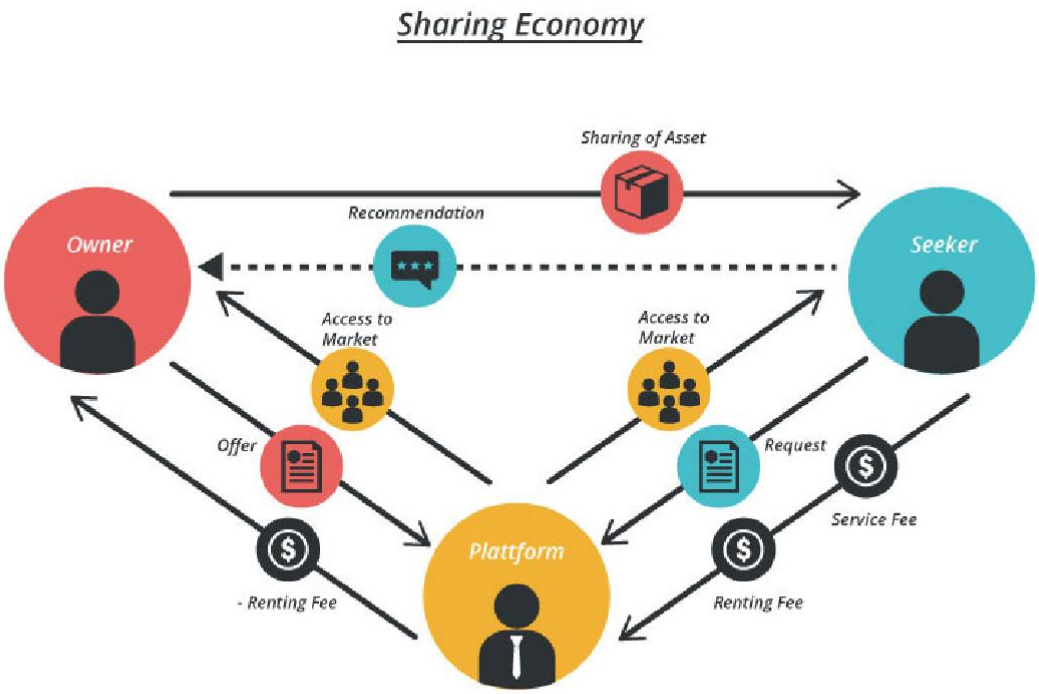
Marketplace Ideation

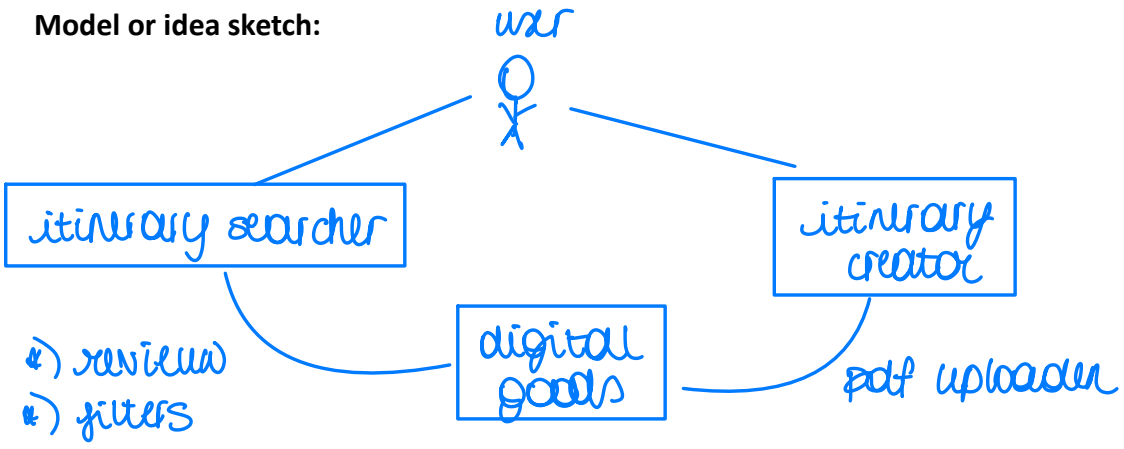
https://www.sharetribe.com/academy/how-to-come-up-with-a-great-marketplace-idea/



Platform name: TravelTales

Short description: TravelTales is een website voor het uploaden en zoeken van reizen. Het is bedoeld als een alternatief voor dure reisbureaus, waarbij je bij TravelTales een heel groot aanbod hebt van verschillende soorten reizen over bijna alle landen van de wereld voor een zeer betaalbare prijs. Door eenvoudigweg een pdf te uploaden waarin al je tips en tricks staan van je reisbestemming, kan je mensen over de hele werd inspireren.
"Share your journey, inspire others"

Model or idea sketch:



Select Properties in the Taxonomy

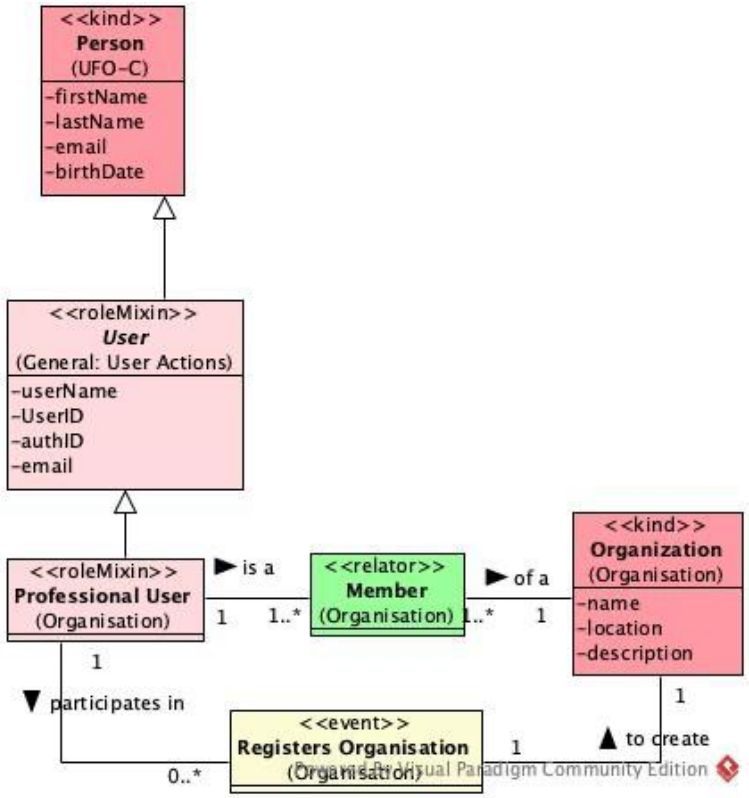
Dimension	Value			
User Type[m]	Person		Organization	
Listing Kind[m]	Good Transfer		Service	
Listing Type[m]	Physical Good[d]	Digital Good[d]	Offline Service[d]	Online Service[d]
Frequency			One-Time[e, d]	Recurring[e, d]
Quantity[m]	One[e]		Many[e]	
Price Discovery	Set by Provider[e]	Set by Customer[e]	Set by Market[e]	
Price Calculation	By Quantity[d]	By Feature[d]	Auction[e, d]	Quote[e, d]
Conversation System	Listing Conversation		Transaction Conversation	
Review by	By Customer		By Provider	
Review of	Of Listing[e, d]	Of Provider[e, d]	Of Customer	
Trust and Safety	ID Verification	Badges	In-app reporting	
Revenue Stream	Subscription	Commission	Fixed Fee	Listing Fee
Revenue Source	Customer		Provider[d]	

Mandatory: 'm', Exclusive: 'e', Dependent: 'd' and thick boxes, grey shading: no ontology modules (yet)

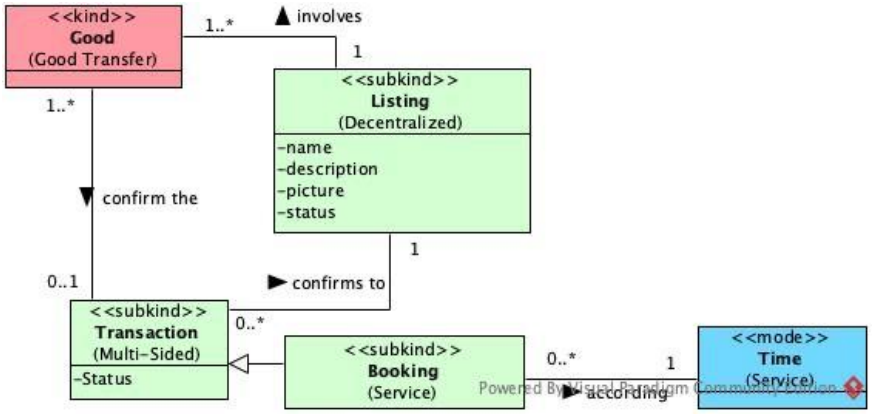
Develop Platform-Specific Ontology

1. Install Visual Paradigm Community Edition (CE): https://www.visual-paradigm.com/download/community.jsp
2. Download the ZIP file ontouml-vp-plugin on https://github.com/OntoUML/ontouml-vp-plugin/releases and install in Visual Paradigm CE via *Help > Install Plugin*
3. Download the Digital Platform Ontology on Github https://github.com/tdrave/Digital_Platform_Ontology and open the *Digital Platform Ontology (DPO+).vpp* file in Visual Paradigm CE
4. Drag and drop relevant classes in the ontology based on the properties selected in the taxonomy (with Thomas)
5. Ctrl+a, right click on a class, *presentation options => show owner => show name only*
6. Reorder, rename and /or delete classes and relationships
7. Add/delete attributes

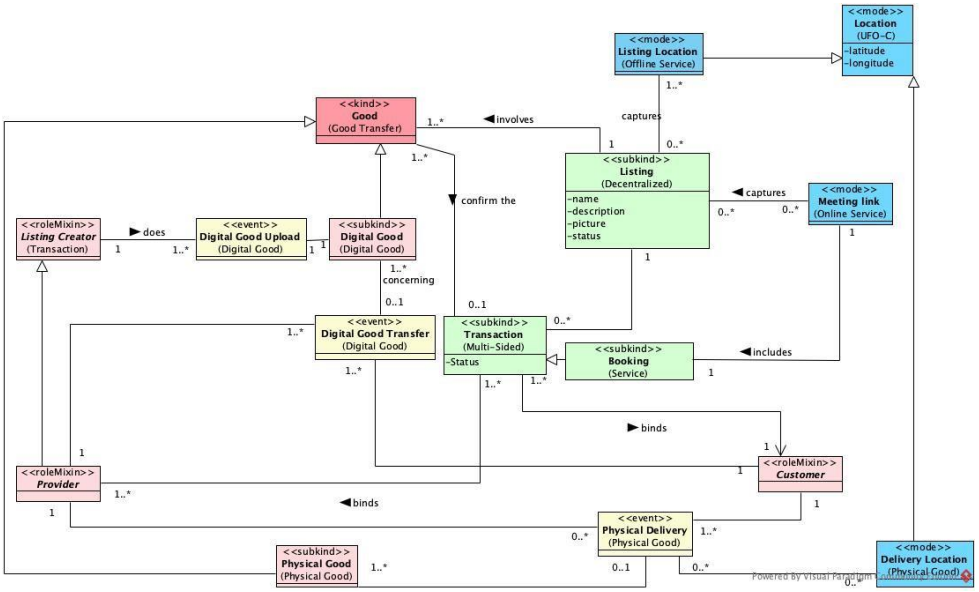
User Type[m]	Person	Organization



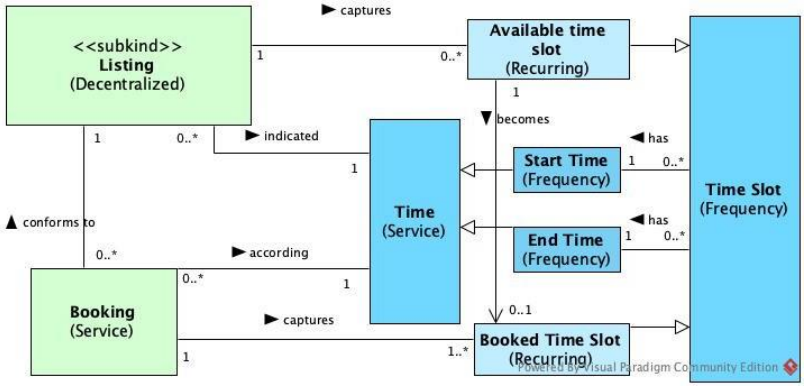
Listing Kind[m]	Good Transfer	Service



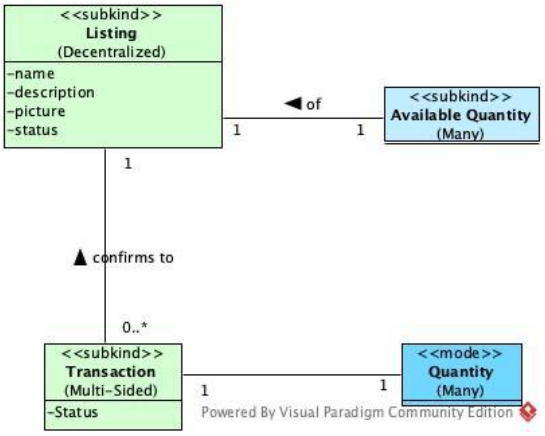
Listing Type[m]	Physical Good[d]	Digital Good[d]	Offline Service[d]	Online Service[d]



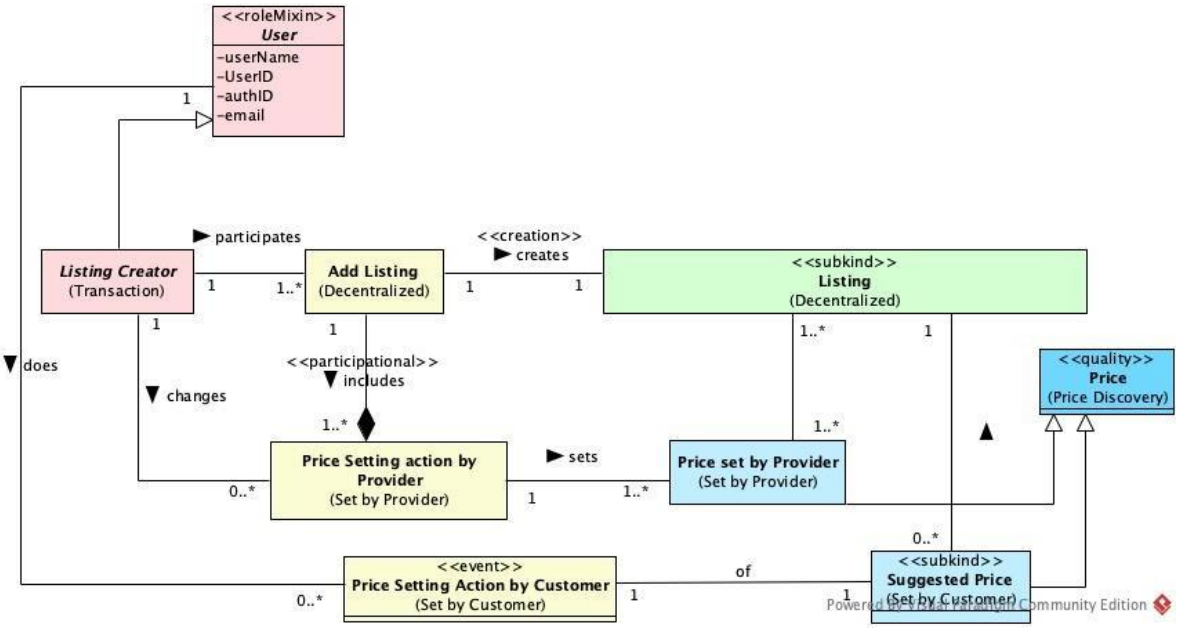
Frequency		One-Time[e, d]	Recurring[e, d]



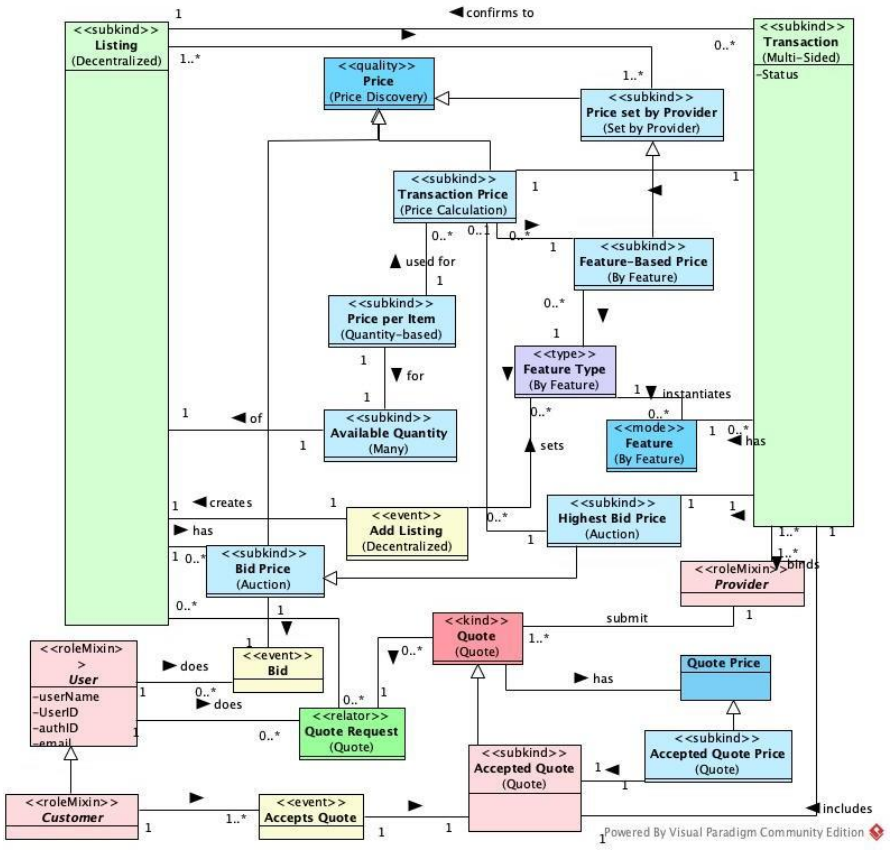
Quantity[m]	One[e]	Many[e]



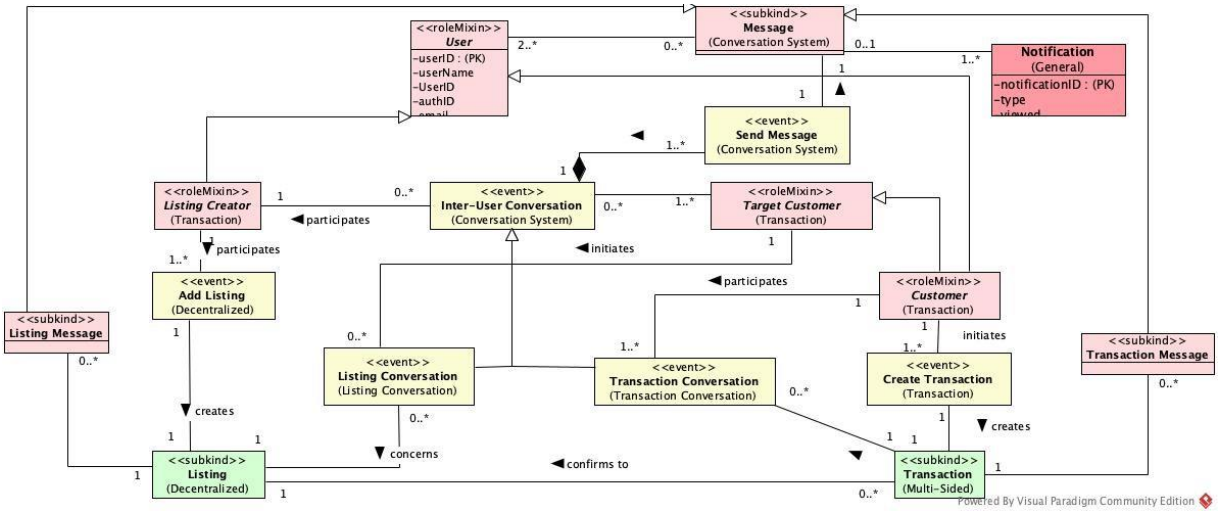
Price Discovery	Set by Provider[e]	Set by Customer[e]	Set by Market[e]



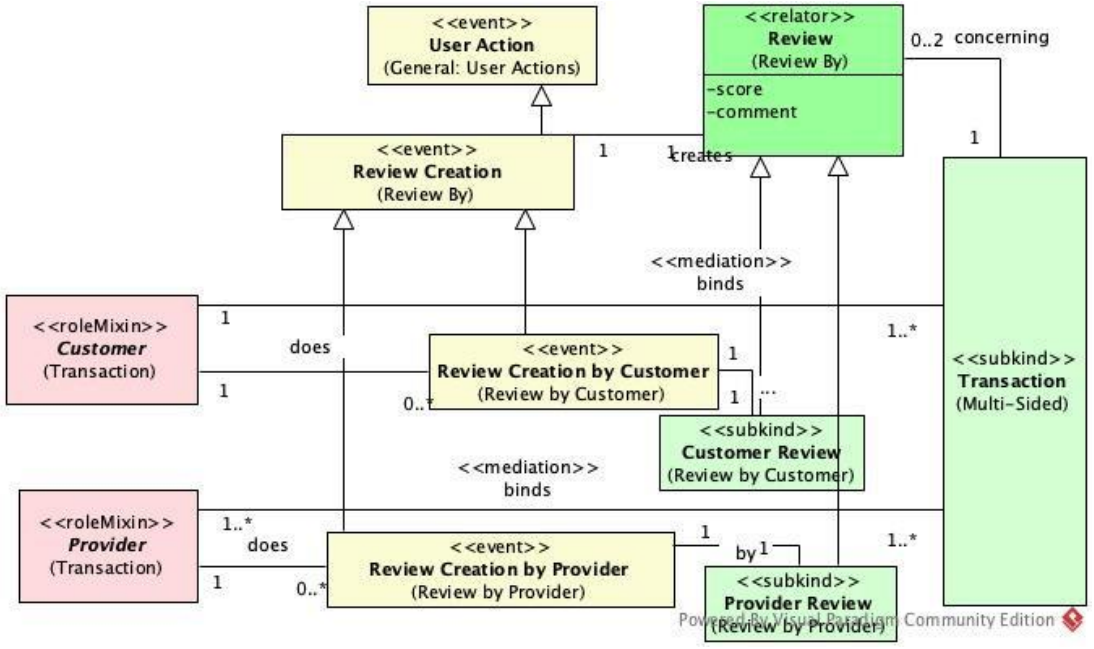
Price Calculation	By Quantity[d]	By Feature[d]	Auction[e, d]	Quote[e, d]



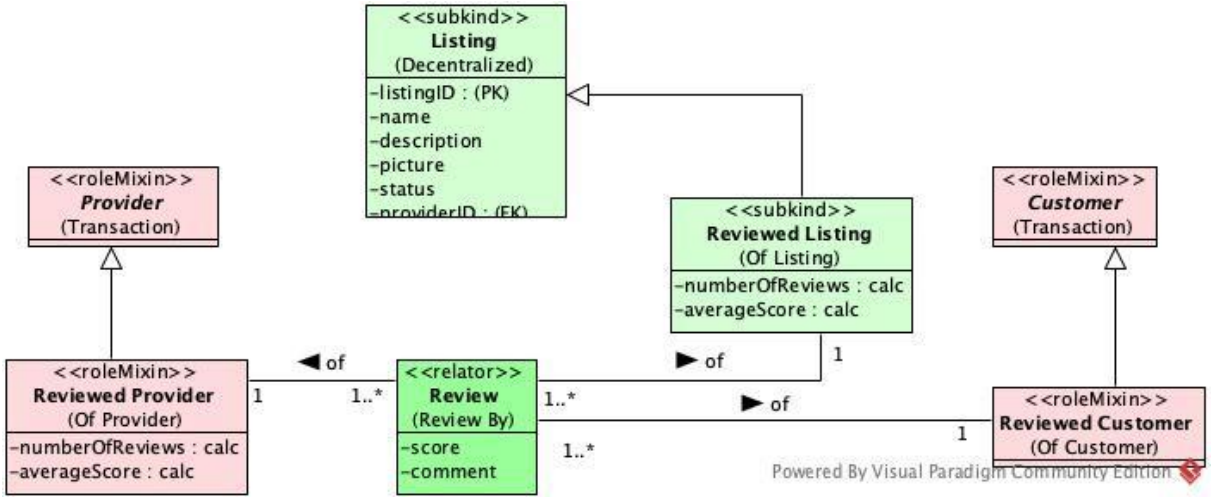
Conversation System	Listing Conversation	Transaction Conversation



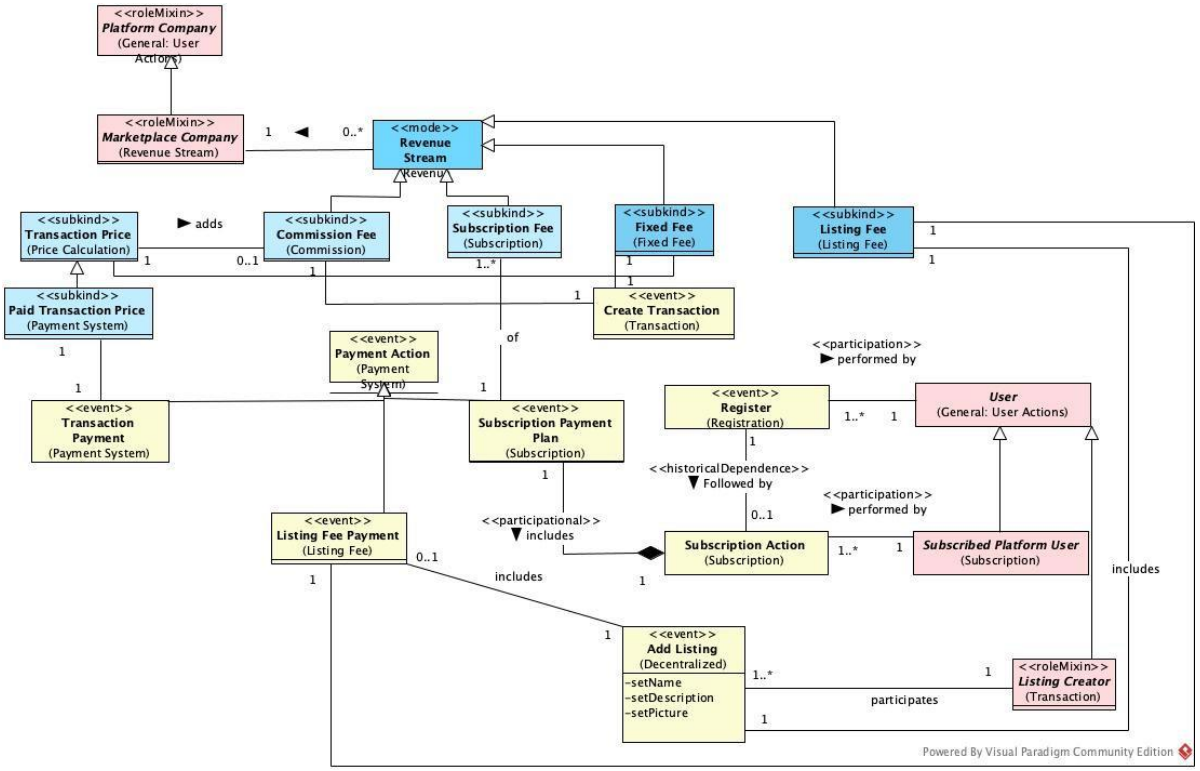
Review by	By Customer	By Provider



Review of	Of Listing[e, d]	Of Provider[e, d]	Of Customer



Revenue Stream	Subscription	Commission	Fixed Fee	Listing Fee

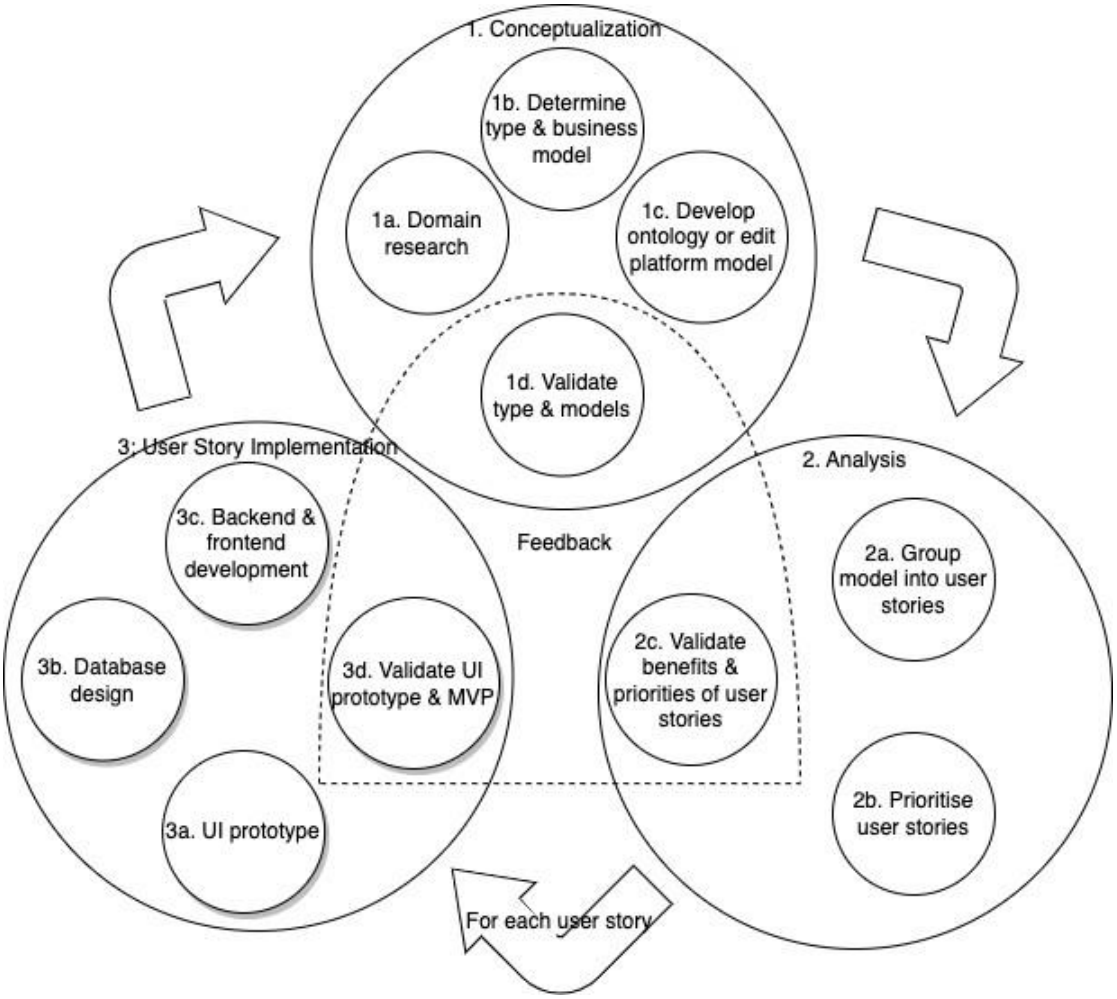


Write User Stories

As a (role) *Red class*	**I want to (goal)** *yellow class*	**So that (benefit)**	**Prio**
Platform bezoeker	Registeren	Ik een gebruiker ben	1
User	Reizen zoeken	Een reis vinden	4
user	Reizen al een pdf uploaden	Andere mensen mijn reizen kunnen kopen	3
User	Login	Zelf een reis kan uploaden	2
User	Favorieten maken	Ik een favorieten reis heb	7
User	Berichten sturen met andere users	Kan communiceren over de reis	16
User	Reis kopen	Pdf kan dowloaden	5
user	Filteren op basis van een weergave op de kaart en op basis van verscheiden filters	Sneller de juiste reis heb	9
user	Review geven op reizen	Andere users meer info hebben over de reis die wordt aangeboden	6
Website eigenaar	Op elke reis die wordt gekocht een commissie van 10%	Winst kunnen maken	15
user	Eigen profiel bewerken & verwijderen	Profiel zoveel aanpassen als hij/zij wil	10
User	Reizen verwijderen	Wanneer ik niet tevreden ben over mijn reis hem gemakkelijk kan verwijderen	12

user	Meldingen krijgen	Ik weet wanneer iemand mijn reis koopt Ik weet wanneer iemand van mijn vrienden een nieuwe reis heeft geplaatst	11
user	Inkomsten & uitgaven bekijken	Ik weet hoeveel ik al verdient hebt en ook hoeveel ik nog kan uitgeven aan reizen	8
user	Andere profielen bekijken en hun geüploade reizen	Vrienden kan toevoegen op basis van hun geüploade reizen	13
User	Geüploade reizen op de kaart zien	Kan ik zien waar ik al geweest ben	17
user	Eigen reviews bekijken	Mijn reizen aanpassen aan de belangen van de zoekers	14

Ontology-driven MVP development method

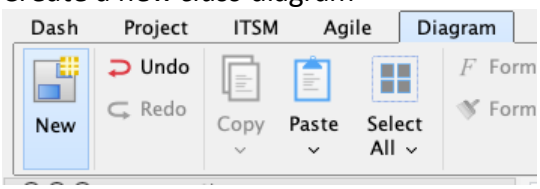


3a: UI Prototype

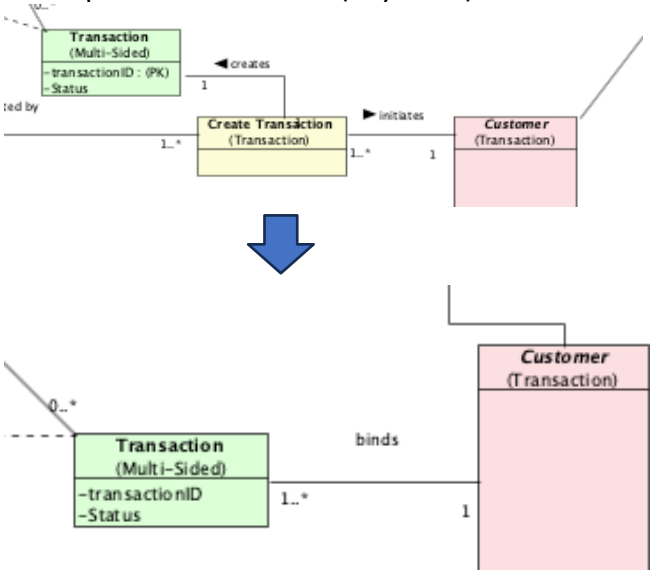
www.figma.com
https://help.figma.com/hc/en-us/categories/360002051613-Get-started
https://www.youtube.com/watch?v=HZuk6Wkx_Eg

3b: From ontology to Entity-Relationship Diagram (ERD): Database Design

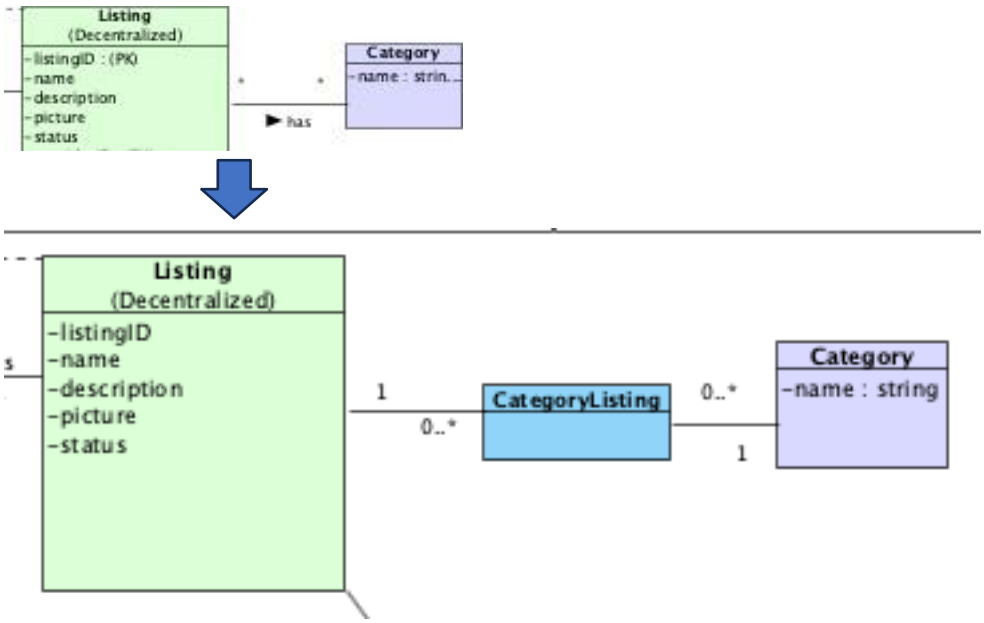
1. Create a new *class* diagram



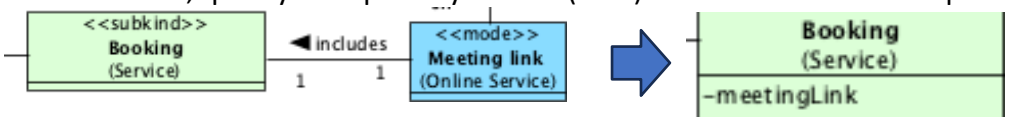
2. Replace event classes (in yellow) with relationships where needed



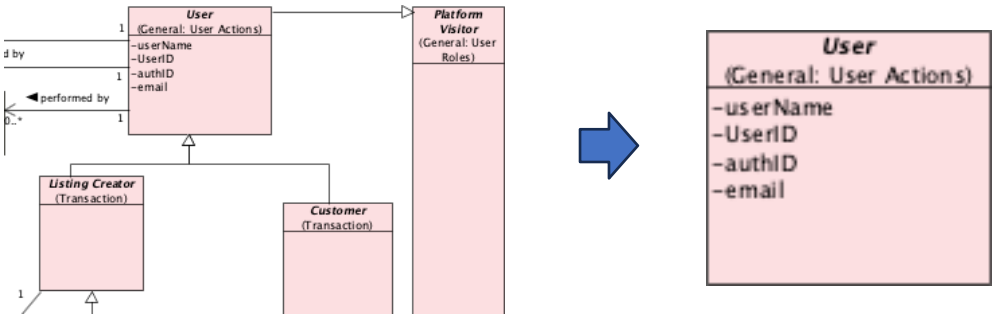
3. Introduce a new table between * to * relationship



4. Add mode, quality and quantity classes (blue) with a 1 – 1 relationship as attributes

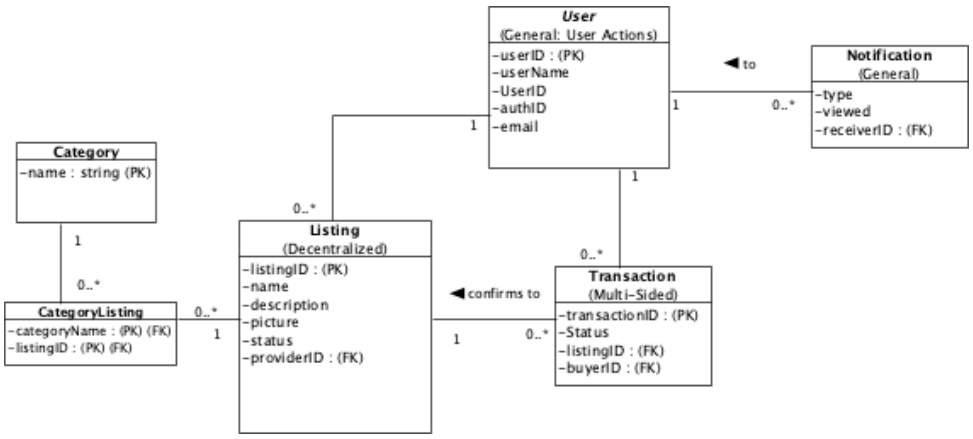


5. Solve each generalization-specialization relationship by changing super - subclasses into
    a. One table for **all** classes (both superclass and subclasses)
    b. One table for **each subclass** (not for the superclass)
    c. One table for **each** class (super and subclasses)



6. Delete enumerations
7. Add Primary Keys (One for each table) and foreign keys (One for each relationship)
8. Optional: Set background style to white (*Styles and Formatting*) and hide owner (*Presentation Options*)
9. Optional: Add datatypes (text, number, date)
10. Optional: Hide FKs in your ontology model (*Selection => Hide*)

Result:



3c: From Ontology to UI Prototype and web application: Backend & frontend development

These steps should be executed in parallel. Focus on the user stories with highest prioritization (Prio):

- Create a screen in your UI prototype (e.g., Figma) for each event (marked in yellow) in your ontology.
- In models.py, create a class for each entity in the Entity-Relationship Diagram (ERD)

```
class Listing(db.Model):
    id = db.Column(db.Integer, primary_key=True)
    listing_name = db.Column(db.String(100), nullable=False)
    price = db.Column(db.Float, nullable=False)
    user_id = db.Column(db.Integer, db.ForeignKey('user.id'), nullable=False)
```

- For each relevant event (marked in yellow) in your ontology, create an HTML template based on the screen of you UI prototype

- In *routes.py,* create a route for each relevant relationship between an event (marked in yellow) and user role (marked in red) in the ontology

```
@main.route('/listings')
def listings():
    all_listings = Listing.query.all()
    return render_template('listings.html', listings=all_listings)
```

As you modify your web application in Flask, continuously update the ontology to reflect the latest changes. Ensure the database aligns with the evolving design and work Agile!