



CrypTour

A Tour Marketplace Powered By The Future



Gliederung

- Generelle Idee
- Konzept: Dezentrale Börse
- Anforderungen
- Vorgehensweise
- Vorstellung Backend
 - Crypto
- Vorstellung Frontend
- Produktdemo

Projektidee

Tourenportal

- Ermöglicht den Kauf und Verkauf von Touren
- Anzeigen dieser Touren

Alleinstellungsmerkmale:

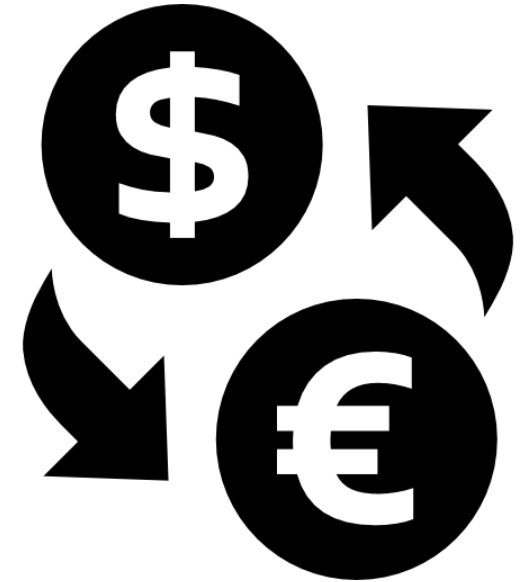
- Jede Tour stellt einen Token dar
- Dezentrale Börse ermöglicht durch:
 - Liquiditätspool
 - Automated Market Maker
- Integration der Wallet durch MetaMask



Dezentrale Börse

Währungsaustausch benötigt:

- Liquidität
 - Klassisch: Market Maker
 - Zentralisiert
 - Ziel: Dezentrale Bereitstellung von Liquidität
- System zur Preiseinigung
 - Klassisch: Konstante Preiseinigung
 - Hohe Anzahl an Aktionen
 - Alternative: Server zur Preiseinigung, Blockchain für Transaktionen
 - Zentralisiert
 - Ziel: Ein einziger Preis, automatisch bestimmt

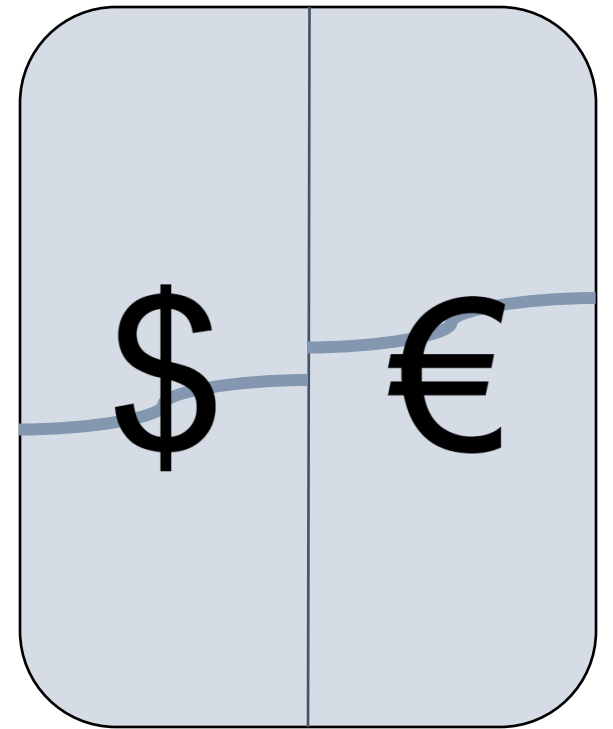


Liquidity Pools

Ziel: Dezentrale Bereitstellung von Liquidität

Lösung

- Ort, an dem Besitzer ihre Tokens ausleihen können
- Gegenleistung: Anteil an Transaktionsgebühren
- Pool sorgt für Liquidität der Börse



Automated Market Maker

Ziel: Ein einziger Preis

Lösung

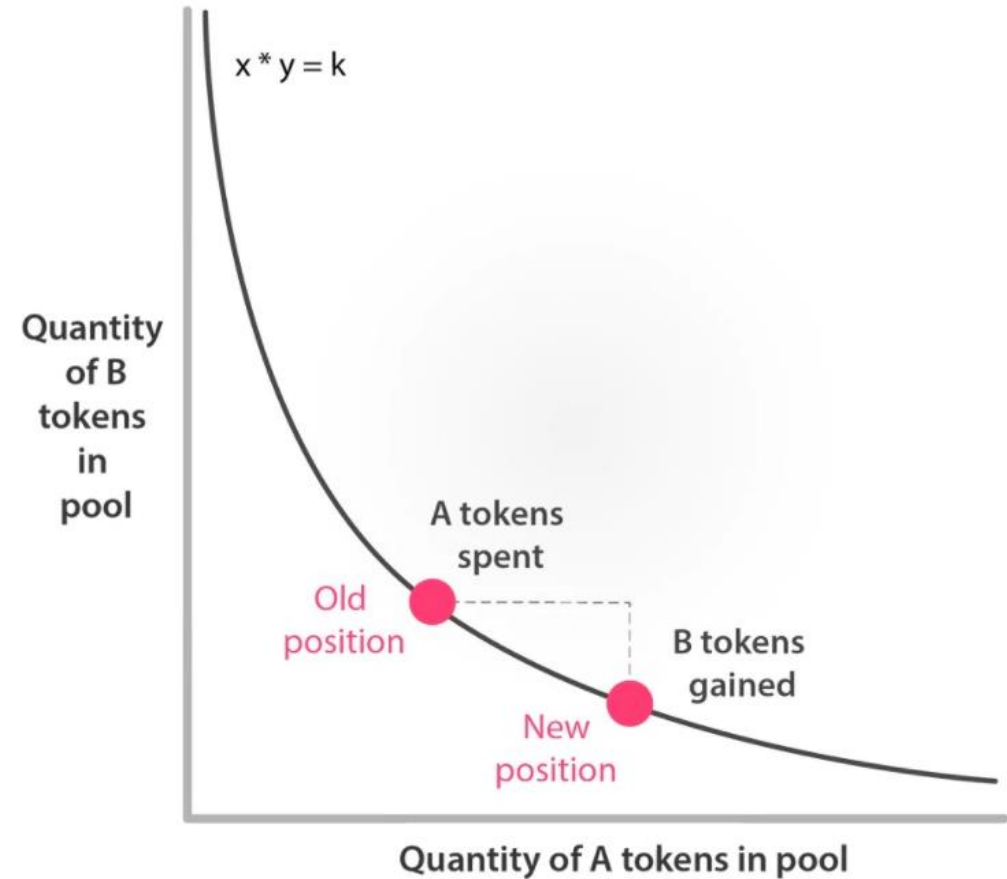
- Preissetzung durch Fixieren von:

$$X * Y = k$$

X: Anzahl erstes Asset

Y: Anzahl zweites Asset

k: Konstante



Anforderungen - Allgemein

Funktionsfähig in Chromium-basierenden Browsern



Kauf von Touren via Tokens

Benutzerfreundlicher Kaufprozess auch für unerfahrene Benutzer

Anforderungen - Tour

- Erstellung
 - Route als GPX-Datei
 - Manuelles setzen von weiteren Informationen
 - Hinzufügen von Bildern
- Darstellung einer Route in einer Karte
- Suche mit zusätzlichen Filtern
- Bewerten von Touren

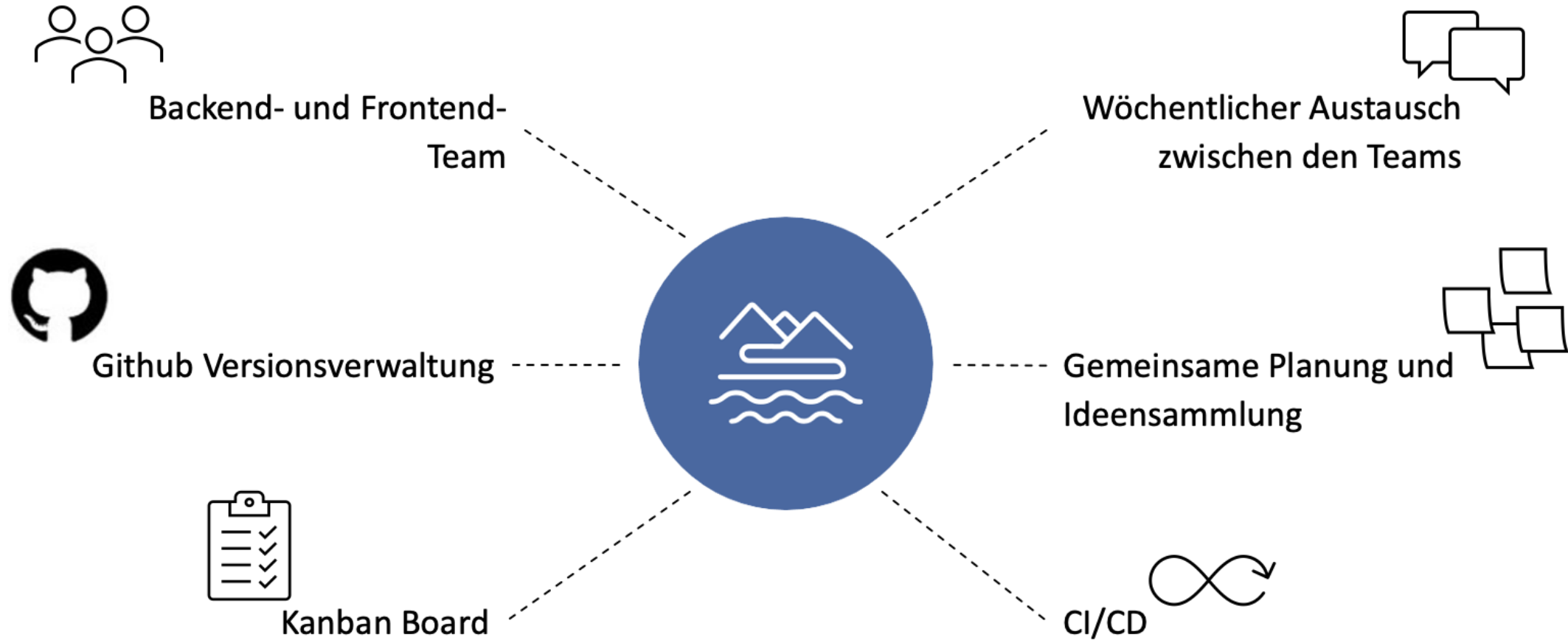


Anforderungen - Benutzerprofile

- Registrierung per E-Mail und eindeutigen Benutzernamen
- Verwaltung
 - gekauften Touren
 - Erstellte Touren
 - Profildaten



Vorgehensweise



Frontend implementieren

Updated now

Filter cards

7 To do

+ ...

Mehr Beispiel Touren hinzufügen

Added by Phlyp

Tour Detailseite schön machen

#9 opened by daullmer

Touren löschen

#3 opened by Phlyp

Touren Bewertung implementieren

Added by Phlyp

Konto: Gekaufte Touren

Added by Phlyp

Konto: User Daten

Automated as To do

Manage

6 In progress

+ ...

filter für die Tourensuche

#6 opened by KaiR-C

Konto: Erstellte Touren

#4 opened by vxccx

eigene touren erstellen

#15 opened by vxccx

Touren mit Backend verbinden

Added by Phlyp

Konto: Crypto Zeug und Zahlungen

Added by Phlyp

Tour erstellen: Crypto-Token Prozess

Automated as In progress

Manage

4 Done

+ ...

User registrieren implementieren

#13 opened by fabiankuffer

1 linked pull request

User login implementieren

#8 opened by fabiankuffer

1 linked pull request

Website in der BW-Cloud hosten

#5 opened by daullmer

1 linked pull request

Tour Kaufen (mit oder ohne Crypto?)

#2 opened by Phlyp

1 linked pull request

Automated as Done

Manage

The image features a white background with decorative curved lines in the corners. In the top-left and bottom-left corners, there are light green and blue curved lines. In the top-right and bottom-right corners, there are light blue and green curved lines.

Backend

API Definition

- Konzept erstellt mit OpenAPI 3
- JavaScript Umsetzung mit open-source Tool generiert
- Neue Methoden: JavaScript <--> OpenAPI

Tours

Tours are the core Dataset of the API.
They describe a planned hiking route and
can be created, sold and bought through
the planned crypto implementation

GET /tours List all tours

POST /tours Create a tour

GET /tours/{TID} Get a specific tour

Users

A user can be a provider or consumer of a
tour.

GET /user Get info about myself

POST /user Create a new User

PATCH /user Edit the information of an existing user

POST

/user Create a new User

Parameters

Try it out

No parameters

Request body required

application/json

The user information

Example Value | Schema

```
{
  "firstname": "string",
  "surname": "string",
  "username": "string",
  "password": "string",
  "email": "user@example.com",
  "wallet-id": "string"
}
```

Responses

Code	Description	Links
201	User successfully created	No links

Media type

application/json

Controls Accept header.

Example Value | Schema

```
{
  "id": 0,
  "firstname": "string",
  "surname": "string",
  "username": "string",
  "email": "user@example.com",
  "walletID": "string"
}
```

DELETE

/tours/{TID} Delete a specific tour

Parameters

Try it out

Name	Description
TID <small>* required</small>	The ID of the tour to delete
integer(\$int64) (path)	<div>TID</div>

Responses

Code	Description	Links
204	Tour deleted	No links
401	Authentication information is missing or invalid	No links

Media type

application/json

Example Value | Schema

```
"Invalid Credentials"
```

| 403 | User does not own tour or tour already bought by someone. Unable to delete. | No links |

Media type

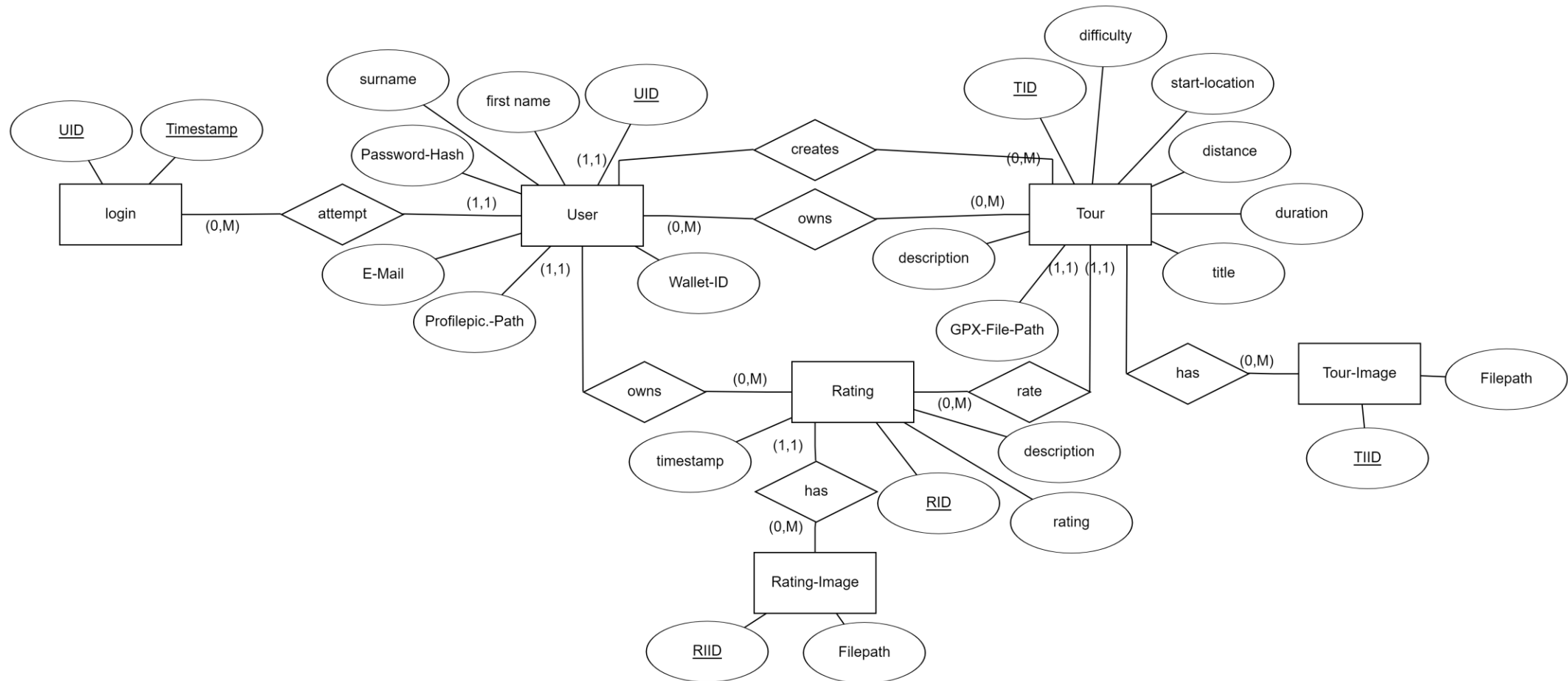
application/json

Example Value | Schema

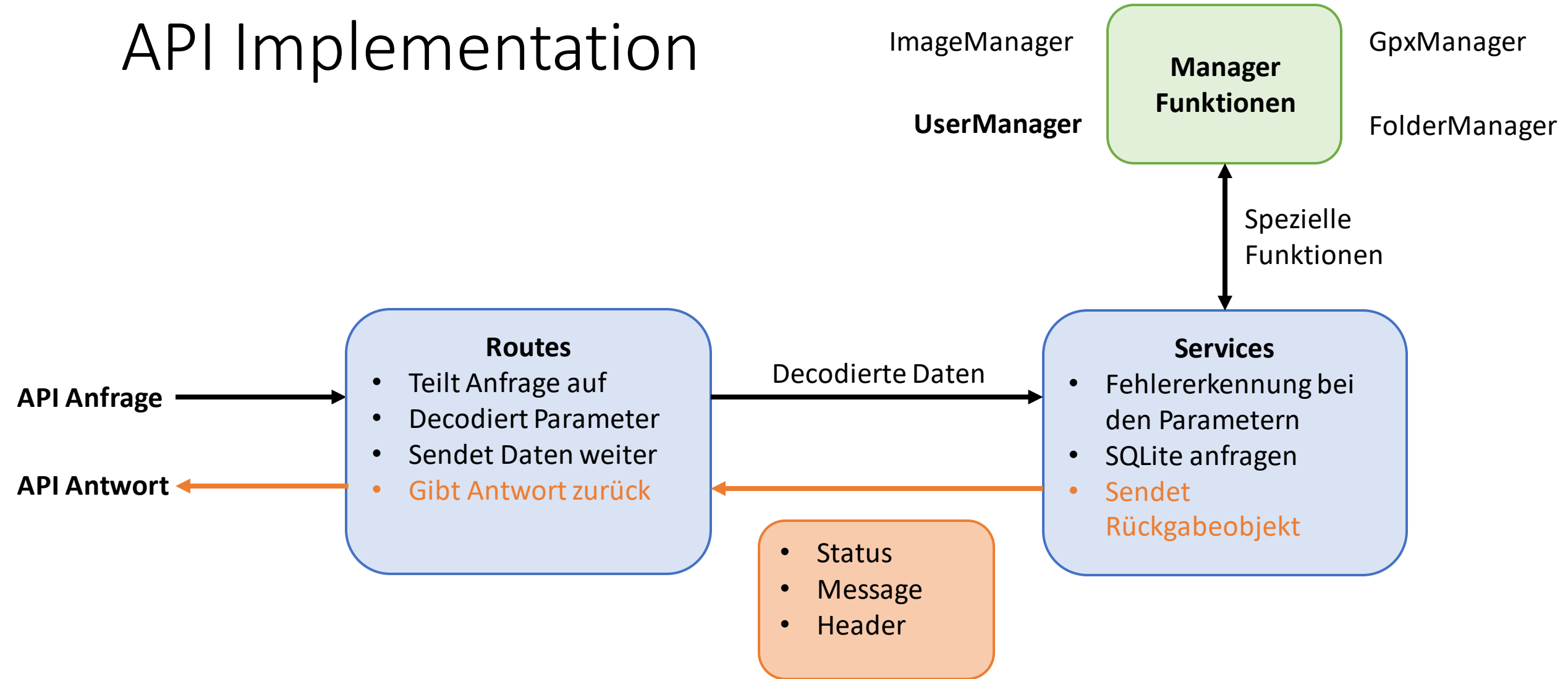
```
{
  "Error": "Tour is already bought and therefore cannot be deleted"
}
```

API Implementation

- Speichern der Daten mit SQLite



API Implementation





Testen mit Postman

SWE / toursOnline

Save

...



GET ▼ `{{url}}/tours?maxDifficulty=1`

Send ▼

Params ● Authorization Headers (7) Body Pre-request Script Tests Settings



Cookies

Query Params

	KEY	VALUE	DESCRIPTION	...	Bulk Edit
<input checked="" type="checkbox"/>	maxDifficulty	1			
	Key	Value	Description		

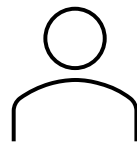
Body Cookies (1) Headers (11) Test Results

Status: 200 OK Time: 315 ms Size: 1.56 KB Save Response ▼

Pretty Raw Preview Visualize JSON ▼ 

```
1 [
2   {
3     "tID": 2,
4     "description": "Einfache Wanderung durch Esslingen am Neckar: Über Obstwiesen und Weinberge zu einem Aussichtspunkt über dem
      Neckartal mit herrlichem Panorama über die Schwäbische Alb.",
5     "title": "Bei der Katharinenlinde",
6     "duration": 180,
7     "distance": 11,
8     "difficulty": 1,
9     "location": "Esslingen",
```

Crypto-Integration



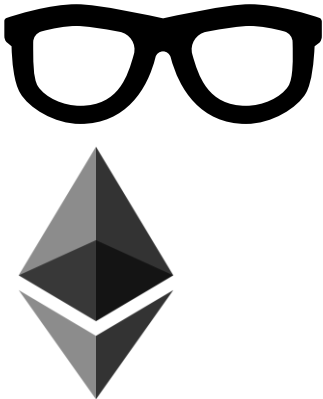
tID	tokenAddress	bpoolAddress
1	0x41E054618b5d6bdF91cB E61e058c4EcbA13ee708	0xEcdb9db04ee1CAe30FAf Ef14a5d88b9C7c26a47f
...

uID	walletAddress
1	0xb91d57743a 2E6D45b9577e 8e32F825dD59 778538

{
 consumer: '0xb91d57743a2E6D45b9577e8e32F825dD59778538',
 payer: '0xb91d57743a2E6D45b9577e8e32F825dD59778538',
 amount: '1000000000000000000',
 serviceId: '1',
 timestamp: '1652364077',
 mrktFeeCollector: '0xb91d57743a2E6D45b9577e8e32F825dD59778538',
 marketFee: '10000000000000000'
}



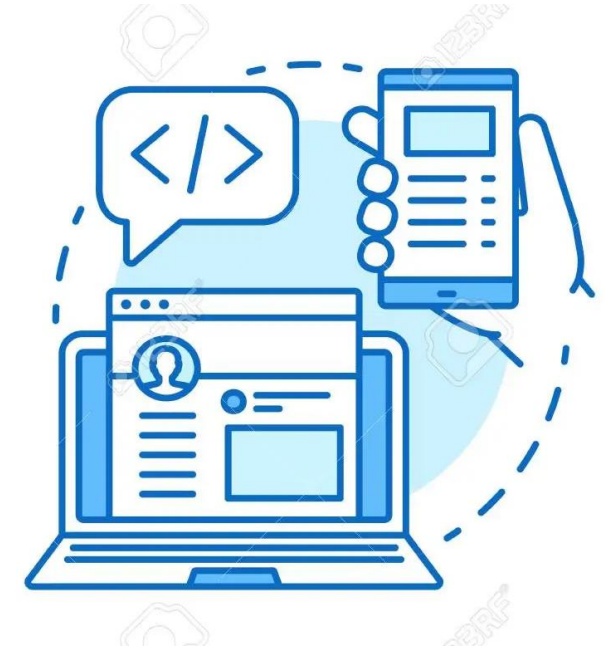
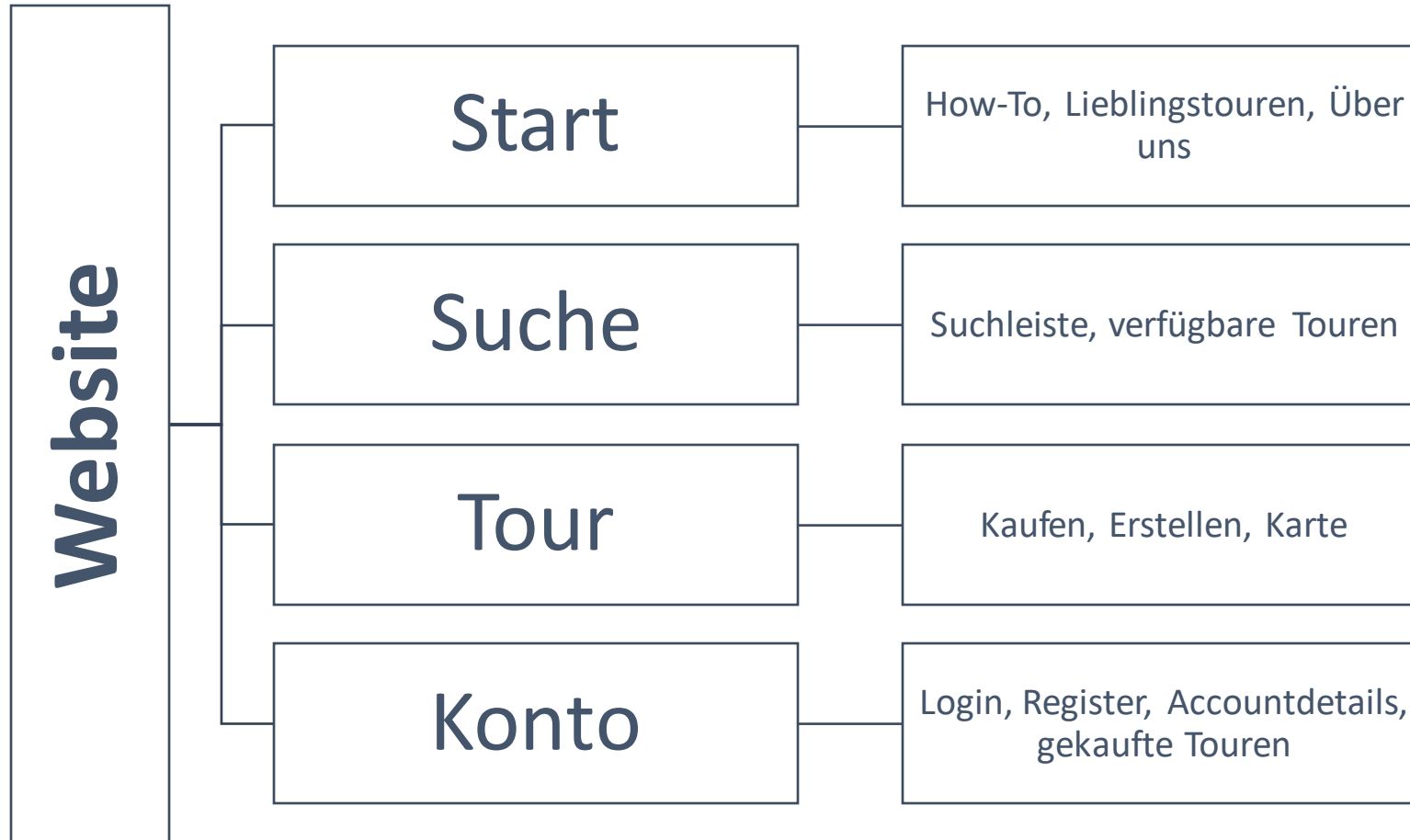
uID	tID
1	1



The image features a light gray background with decorative curved lines in the corners. In the top-left and bottom-left corners, there are thick, multi-layered curved lines in shades of light blue and green. In the top-right corner, there is a similar thick, multi-layered curved line in shades of light blue and green.

Frontend

Frontend





Kurze Produktdemo