

ToTheLoo

Jamal, Jonas, Lena, Niklas und Patrick

Allgemeines zur App

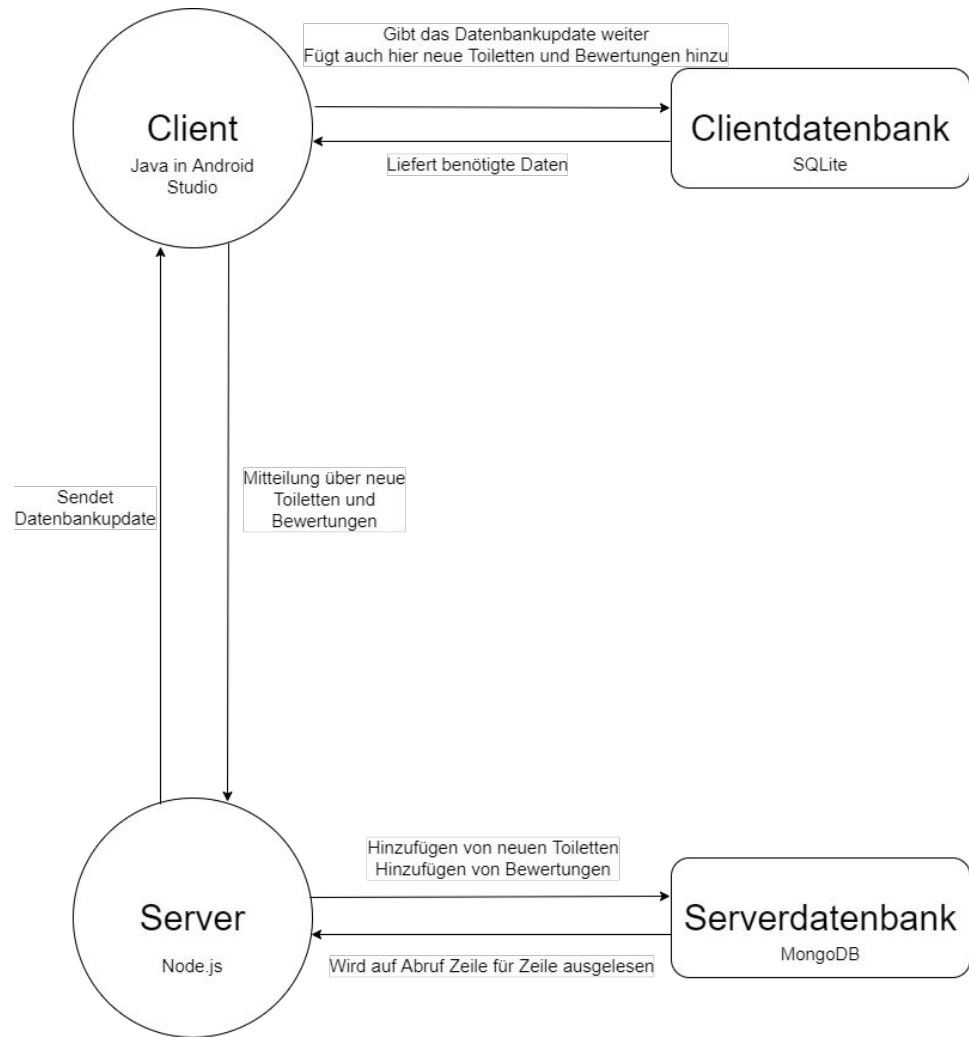
Was soll sie können?

- Anzeige der Toiletten in der Umgebung
- Anzeige von Details einer ausgewählten Toilette
- Routing zu einer ausgewählten Toilette
- Toiletten Reviews erstellen

Nice-to-haves:

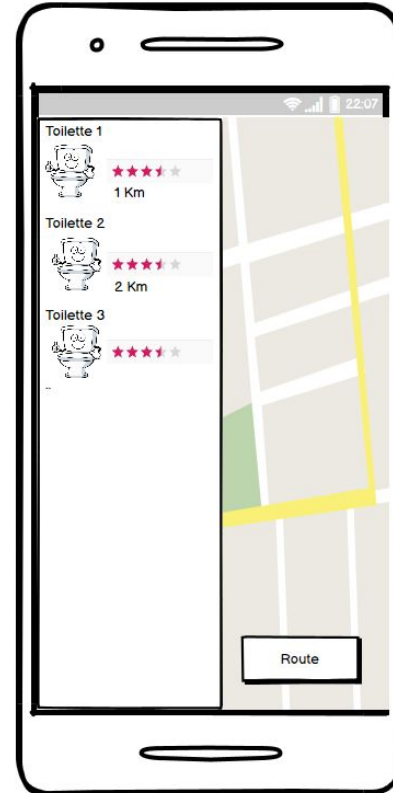
- Filteroptionen für die anzuzeigenden Toiletten
- Hinzufügen neuer Toiletten

Kommunikation



Erste UI-Entwürfe





Kurze Präsentation des aktuellen UI

Klientendatenbank Schema

- Relationales Datenbankmodell

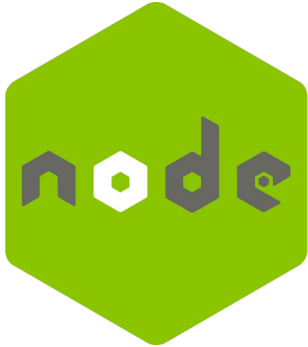
ID, Name, Breitengrad, Längengrad, Tag, Wegbeschreibung, Beschreibung,
Bewertungen(Benutzer, Bewertungstext, Sterne)

Klientendatenbank Implementation

- SQLite
- Vorteil: bereits in Android Studio implementiert
 - ⇒ Erstellung der Datenbank simpel
 - ⇒ Kommunikation mit Datenbank einfach
- Eingabe von Daten als String
- Ausgabe als Cursor Objekt

Aufbau des Backends

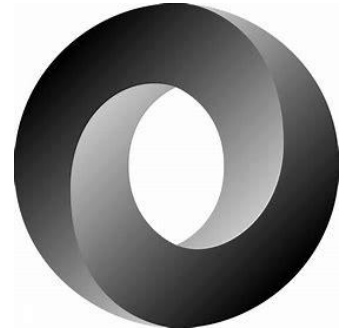
Usecase des Backends: Speicherung von Daten aller Nutzer um eine möglichst große Datenbasis aufzubauen



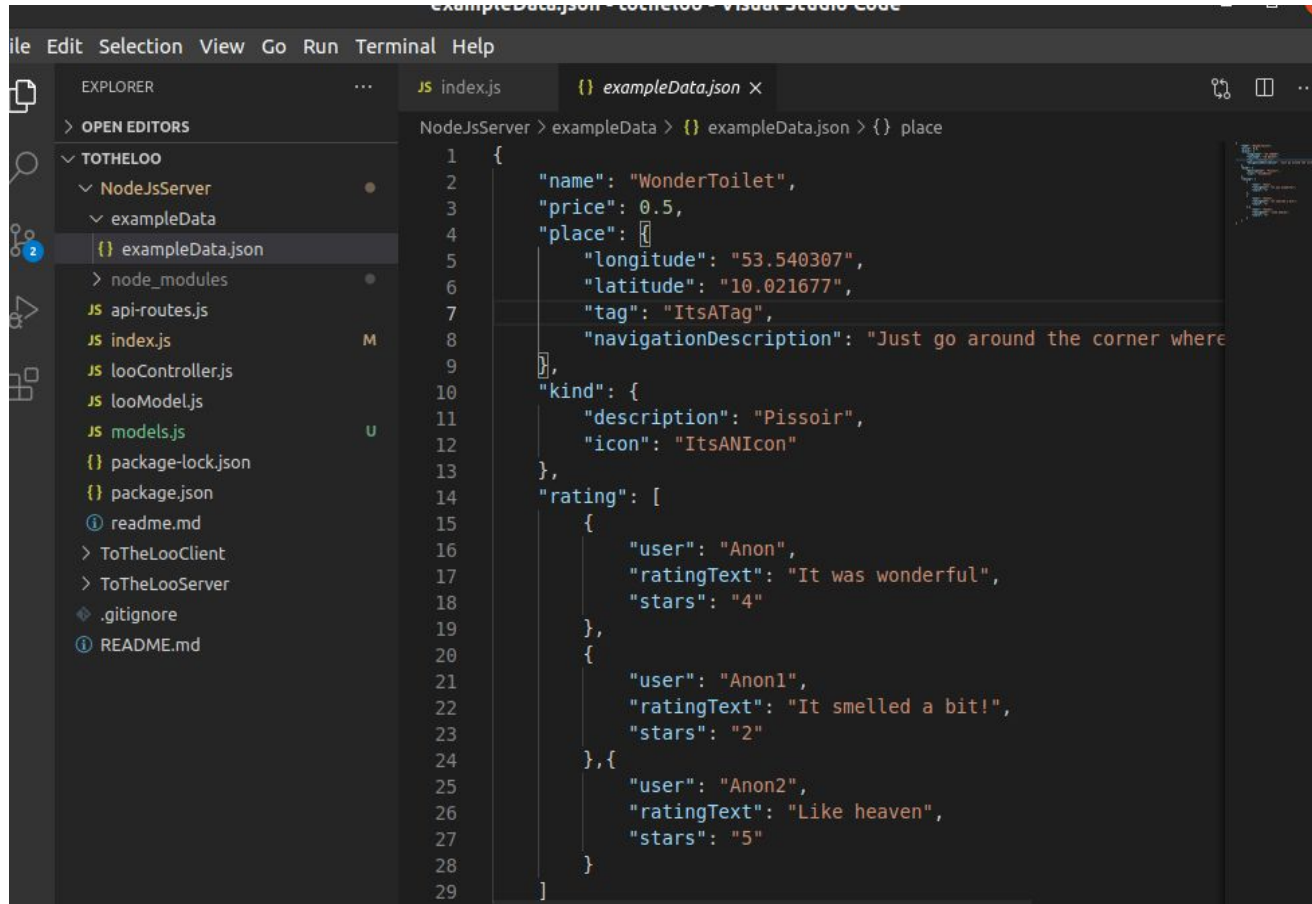
- Javascriptbasiertes Serverframework
- Benutzung von Express-Modul für REST API Erstellung



- Mongo=humongous
- NoSQL - dokumentenbasiert
- verwaltet JSON-Dokumente
- läuft auf der VM

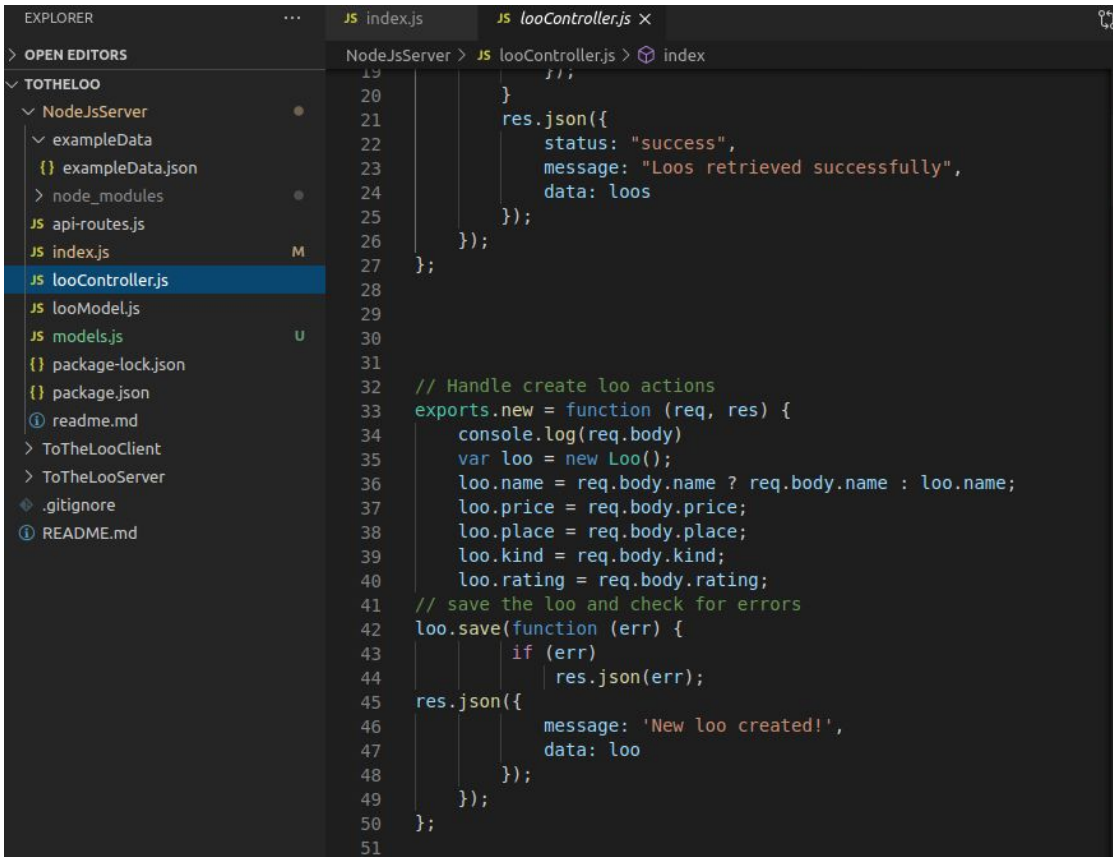


Datenmodell - Momentaner Aufbau eines LooObject



```
1 {
2   "name": "WonderToilet",
3   "price": 0.5,
4   "place": {
5     "longitude": "53.540307",
6     "latitude": "10.021677",
7     "tag": "ItsATag",
8     "navigationDescription": "Just go around the corner where
9   },
10  "kind": {
11    "description": "Pissoir",
12    "icon": "ItsANIcon"
13  },
14  "rating": [
15    {
16      "user": "Anon",
17      "ratingText": "It was wonderful",
18      "stars": "4"
19    },
20    {
21      "user": "Anon1",
22      "ratingText": "It smelled a bit!",
23      "stars": "2"
24    },
25    {
26      "user": "Anon2",
27      "ratingText": "Like heaven",
28      "stars": "5"
29    }
30  ]
31 }
```

Controller - Handling der Requests



The image shows a screenshot of the Visual Studio Code editor. On the left, the Explorer sidebar displays the project structure for 'TOTHELOO'. The 'NodeJsServer' folder is expanded, showing files like 'exampleData.json', 'node_modules', 'api-routes.js', 'index.js', 'looController.js' (which is selected), 'looModel.js', and 'models.js'. The main editor area shows the code for 'looController.js'. The code defines a new Express route handler for the POST endpoint '/loos'. It logs the request body, creates a new 'Loo' object with properties from the request body, saves it to the database using 'loos.save()', and returns a JSON response with a success message and the created loo object.

```
19 // ...  
20 }  
21 res.json({  
22   status: "success",  
23   message: "Loos retrieved successfully",  
24   data: loos  
25 });  
26 }  
27 }  
28  
29  
30  
31  
32 // Handle create loo actions  
33 exports.new = function (req, res) {  
34   console.log(req.body)  
35   var loo = new Loo();  
36   loo.name = req.body.name ? req.body.name : loo.name;  
37   loo.price = req.body.price;  
38   loo.place = req.body.place;  
39   loo.kind = req.body.kind;  
40   loo.rating = req.body.rating;  
41   // save the loo and check for errors  
42   loo.save(function (err) {  
43     if (err)  
44       res.json(err);  
45     res.json({  
46       message: 'New loo created!',  
47       data: loo  
48     });  
49   });  
50 }  
51
```

Bisher eingestellte Routen (Schnittstellen)

The image shows a VS Code editor with the file explorer on the left and the code editor in the center. The file explorer shows the project structure for 'TOTHELOO', with 'api-routes.js' selected. The code editor shows the content of 'api-routes.js', which defines an Express router with a default route and several API endpoints. A browser window is open in the foreground, showing the response from the API at 'localhost:8080/api'.

EXPLORER

- > OPEN EDITORS
- TOTHELOO
 - NodeJsServer
 - exampleData
 - exampleData.json
 - node_modules
 - JS api-routes.js**
 - JS index.js
 - JS looController.js
 - JS looModel.js
 - JS models.js
 - package-lock.json
 - package.json
 - readme.md
 - ToTheLooClient
 - ToTheLooServer
 - .gitignore
 - README.md

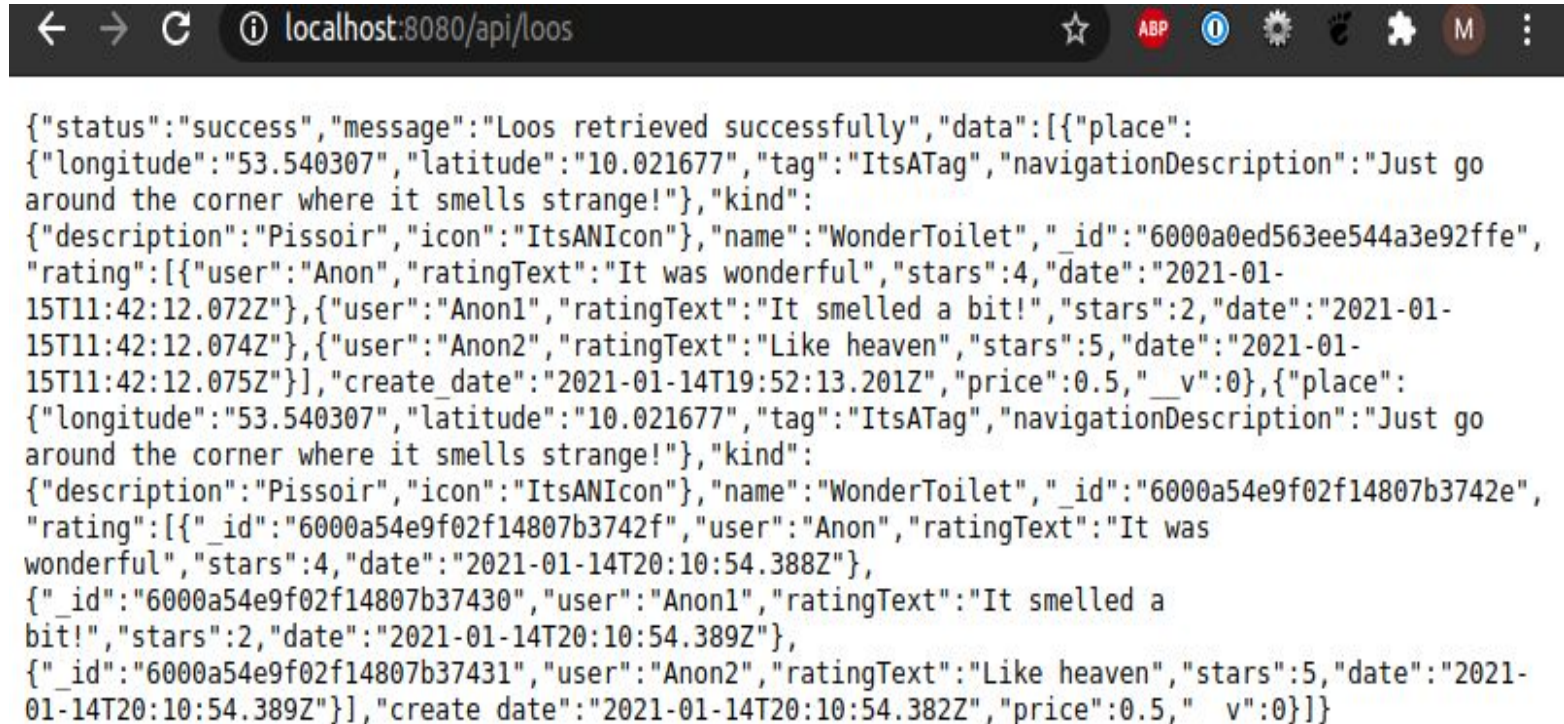
JS api-routes.js

```
1 // Initialize express router
2 let router = require('express').Router();
3 // Set default API response
4 router.get('/', function (req, res) {
5   res.json({
6     status: 'Api is working',
7     message: 'Welcome to the Loo Server. Find the best public
8   });
9 });
10 // Import loo controller
11 var looController = require('./looController');
12 // loo routes
13 router.route('/loos')
14   .get([looController.index])
15   .post(looController.new);
16 router.route('/loos/:loo_id')
17   .get(looController.view)
18   .patch(looController.update)
19   .put(looController.update)
20   .delete(looController.delete);
21 // Export API routes
22 module.exports = router;
```

Browser Preview: localhost:8080/api

```
{
  "status": "Api is working",
  "message": "Welcome to the Loo
  ver. Find the best public loo data!!"}
}
```

Ausgabe des get-Requests auf alle Loos



```
{
  "status": "success",
  "message": "Loos retrieved successfully",
  "data": [
    {
      "place": {
        "longitude": "53.540307",
        "latitude": "10.021677",
        "tag": "ItsATag",
        "navigationDescription": "Just go around the corner where it smells strange!"
      },
      "kind": {
        "description": "Pissoir",
        "icon": "ItsANIcon",
        "name": "WonderToilet",
        "_id": "6000a0ed563ee544a3e92ffe",
        "rating": [
          {
            "user": "Anon",
            "ratingText": "It was wonderful",
            "stars": 4,
            "date": "2021-01-15T11:42:12.072Z"
          },
          {
            "user": "Anon1",
            "ratingText": "It smelled a bit!",
            "stars": 2,
            "date": "2021-01-15T11:42:12.074Z"
          },
          {
            "user": "Anon2",
            "ratingText": "Like heaven",
            "stars": 5,
            "date": "2021-01-15T11:42:12.075Z"
          }
        ],
        "create_date": "2021-01-14T19:52:13.201Z",
        "price": 0.5,
        "__v": 0
      },
      "place": {
        "longitude": "53.540307",
        "latitude": "10.021677",
        "tag": "ItsATag",
        "navigationDescription": "Just go around the corner where it smells strange!"
      },
      "kind": {
        "description": "Pissoir",
        "icon": "ItsANIcon",
        "name": "WonderToilet",
        "_id": "6000a54e9f02f14807b3742e",
        "rating": [
          {
            "_id": "6000a54e9f02f14807b3742f",
            "user": "Anon",
            "ratingText": "It was wonderful",
            "stars": 4,
            "date": "2021-01-14T20:10:54.388Z"
          },
          {
            "_id": "6000a54e9f02f14807b37430",
            "user": "Anon1",
            "ratingText": "It smelled a bit!",
            "stars": 2,
            "date": "2021-01-14T20:10:54.389Z"
          },
          {
            "_id": "6000a54e9f02f14807b37431",
            "user": "Anon2",
            "ratingText": "Like heaven",
            "stars": 5,
            "date": "2021-01-14T20:10:54.389Z"
          }
        ],
        "create_date": "2021-01-14T20:10:54.382Z",
        "price": 0.5,
        "__v": 0
      }
    ]
  }
}
```