**BACKEND:**

1. V datoteki /app/backend/db deklarirate novo entiteto (objekt, ki vam bo vračal podatke iz bazne tabele) glede na vsebino (npr. za breeweries):

Breeweries :bookshelf.Model.extend(

{​​​​​​​​

tableName:'breeweries',

idAttribute:'id'

}​​​​​​​​) (ne pozabiti na vejine, ker je to seznam!)

1. V mapo /app/backend/routers kopirate obstoječo datoteko iz tega imenika (npr. tutorials.js) in jo poimenujete glede na vsebino (npr. breeweries.js)
2. Spremenite parametre funkcije router.get (primer za breeweries): router.get('/breweries', async function(req, res, next) {​​​​​​​     try {​​​​​​​       let users = await new db.entity.Breweries().fetchAll();       res.json(users.toJSON());     }​​​​​​​ catch(error){​​​​​​​         res.status(500).json(error);     }​​​​​​​ }​​​​​​​);
3. Dodate novi router (kontroler) v api.js tako, da ga navedete v seznam routerjev za api.use('/', .. seznam):

api.use('/',

require('./routers/tutorials'),

require('./routers/users'),

**require('./routers/breweries')**

); (pozor! spet pazite na vejice, ker je seznam)

1. S tem smo ustvarili novi backend endpoint, ki ga preizkusite na naslovu: <http://localhost:8000/api/breweries> V kolikor imamo v bazi zapise v tabeli Breweries, bi morali dobiti JSON vseh pivovarn.

**FRONTEND:**

1. V datoteko /app/frontend/frontend.js dodamo novi router (kontroler, usmerjevalnik) v **seznam** routerjev:

frontend.use('/',

require('./routers/tutorials'),

require('./routers/users'),

**require('./routers/breweries')**

);

1. Kopiramo enega izmed obstoječih routerjev in ga preimenujemo: /app/frontend/routers/tutorials.js -- COPY -> /app/frontedn/routers/breweries.js
2. spremenimo parametre funkcije **router.get** tako, da uporabimo prej ustvarjen API endpoint v backend-u: router.get('/breweries', function(req, res, next) {​​​​​​​     let data = ws.get('/breweries').then((response) => {​​​​​​​       res.render('breweries', {​​​​​​​ breweries: response.data }​​​​​​​);     }​​​​​​​).catch((error) => {​​​​​​​       res.status('404').send('Napaka pri pridobivanju pivovarn: ' + error);     }​​​​​​​); }​​​​​​​);
3. Ustvariti moramo novo predlogo spletne strani /app/frontend/views/**breweries.html**
4. Metoda render iz točke 3. bo uporabila to predlogo s podatki, ki smo jih podali: {​​​​​​​ breweries: response.data }​​​​​​​. Na mestu, kjer želimo izpisati elemente, uporabimo Handlebards template engine sintakso (glej: <https://handlebarsjs.com/>). Opomba: Response.data je javascript array. Template engine pričakuje, da dobi Javascript objekt, ne array. Zato "wrap-amo" array v objekt na način, kot je zapisan: <obj>: {​​​​​​​ breweries: <array>}​​​​​​​ Primer izpisa pivovarn v obliki tabele:

<table ...> <thead><tr><th>Naziv</th><th>Dat. odprtja</th></tr></thead> <tbody> **{​​​​​​​{​​​​​​​#each breweries}​​​​​​​}​​​​​​​   <tr><td>{​​​​​​​{​​​​​​​ name }​​​​​​​}​​​​​​​</td><td>{​​​​​​​{​​​​​​​ dat\_opened }​​​​​​​}​​​​​​​</td></tr> {​​​​​​​{​​​​​​​/each}​​​​​​​** </tbody> </table>

1. Preizkusimo spletno stran na naslovu: <http://localhost:8000/breweries>