JavaScript-05
Postman

23 OKTOBER 2024

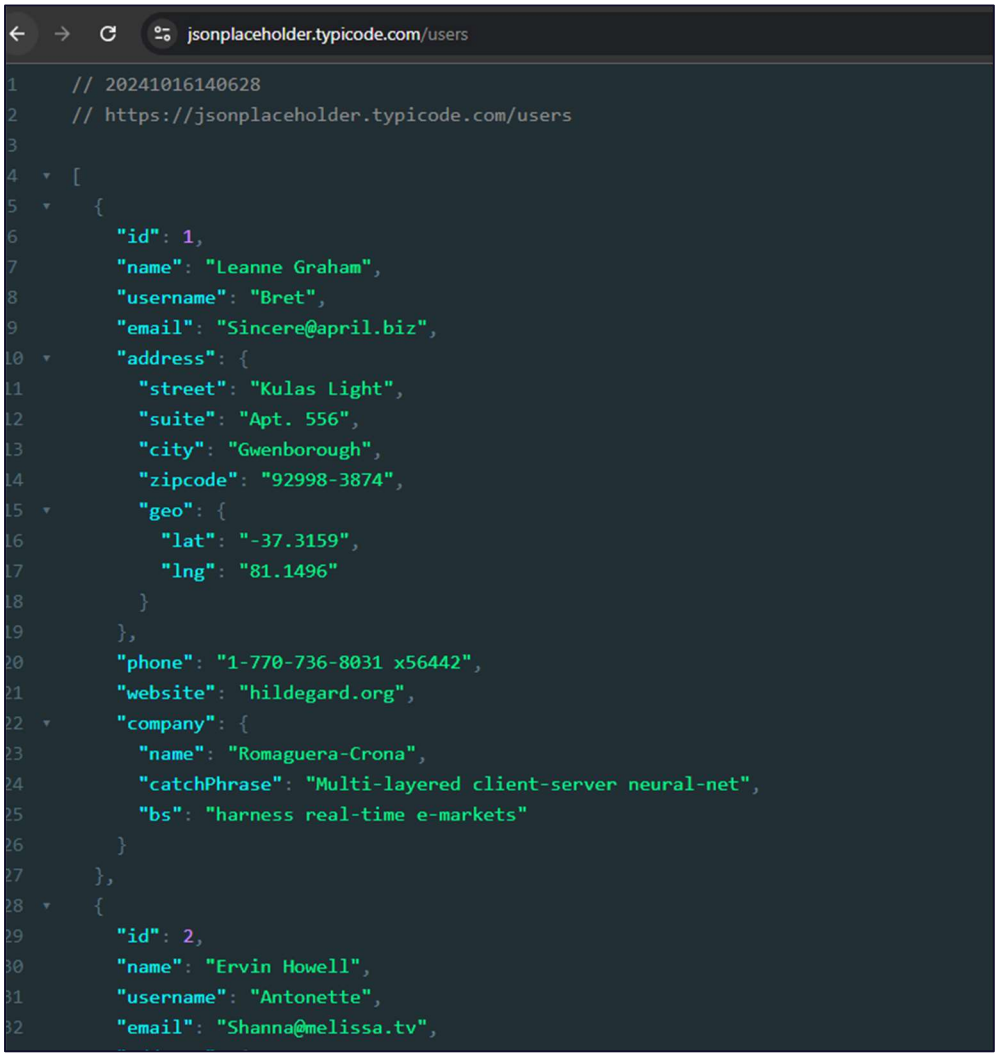
JavaScript – utan tjafs
Författare: Acke Strömberg

För en kanin

Postman

Fakta om Postman

När man utvecklar webbapplikationer och vill testa anrop mot REST-API:er är Postman ett mycket bra verktyg. Man vill gärna se vad man får tillbaka som retur från ett REST API och också testa hur det går att skicka till detsamma. Om man endast behöver köra HTTP request metoden GET så klarar man sig faktiskt utmärkt med bara en vanlig webbläsare som Chrome eller Egde. Du skriver helt enkelt in adressen till ditt REST API och trycker på enter. Då ser det ut som på figur 1 om allting är korrekt. Här är det en GET request till https://jsonplaceholder.typicode.com/users .



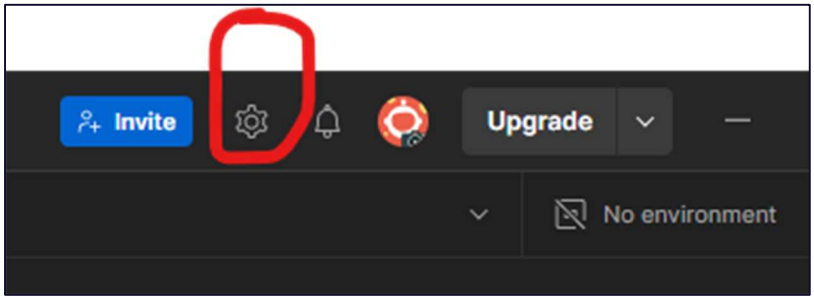
Figur 1 Inställningar uppe i högra hörnet.

Det går däremot inte att göra POST, PUT, PATCH eller DELETE i en webbläsare. För det behöver man Postman för att hantera det.

Ladda ner

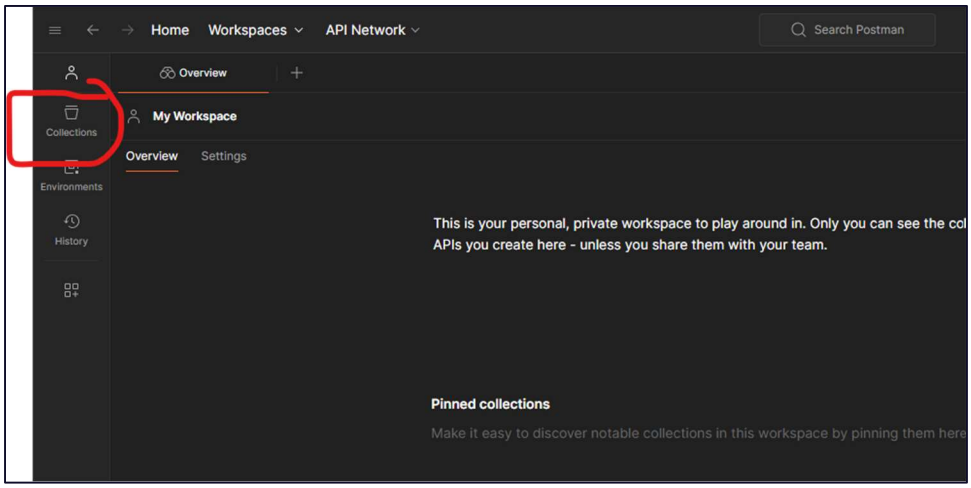
Det går att köra Postman både som webbapplikation och desktop-applikation. För den här manualen används desktop-applikationen. Allting borde se exakt lika ut men det är inte garanterat att det är så. Börja med att gå till länken https://postman.com/downloads/ och ladda ner den versionen som hör till ditt operativsystem. När nedladdningen är färdig så kan du starta applikationen och skapa ett konto. Det är några funktioner som inte går att komma åt om man inte är inloggad. Det kostar ingenting att använda Postman om man kör det som studerande och inte har det som ett företag.

När applikationen har startat kan du gå in på inställningar uppe i högra hörnet, enligt figur 2. Där klickar du på Settings och sen kan du välja ett tema, mörkt eller ljust, eller välja att köra på det som är inställt på din dator.



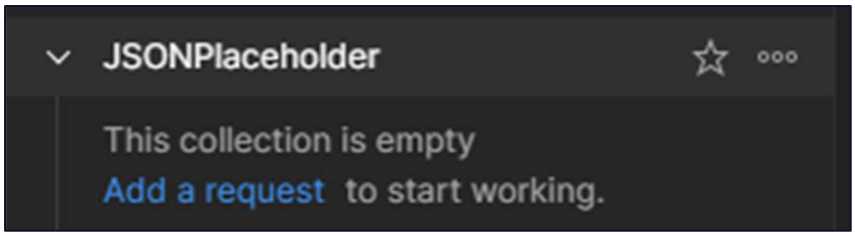
Figur 2 Inställningar uppe i högra hörnet.

Längst till vänster finns det några ikoner. Klicka på den som det står Collections på enligt figur 3.



Figur 3 Ikonen Collections till vänster i applikationen.

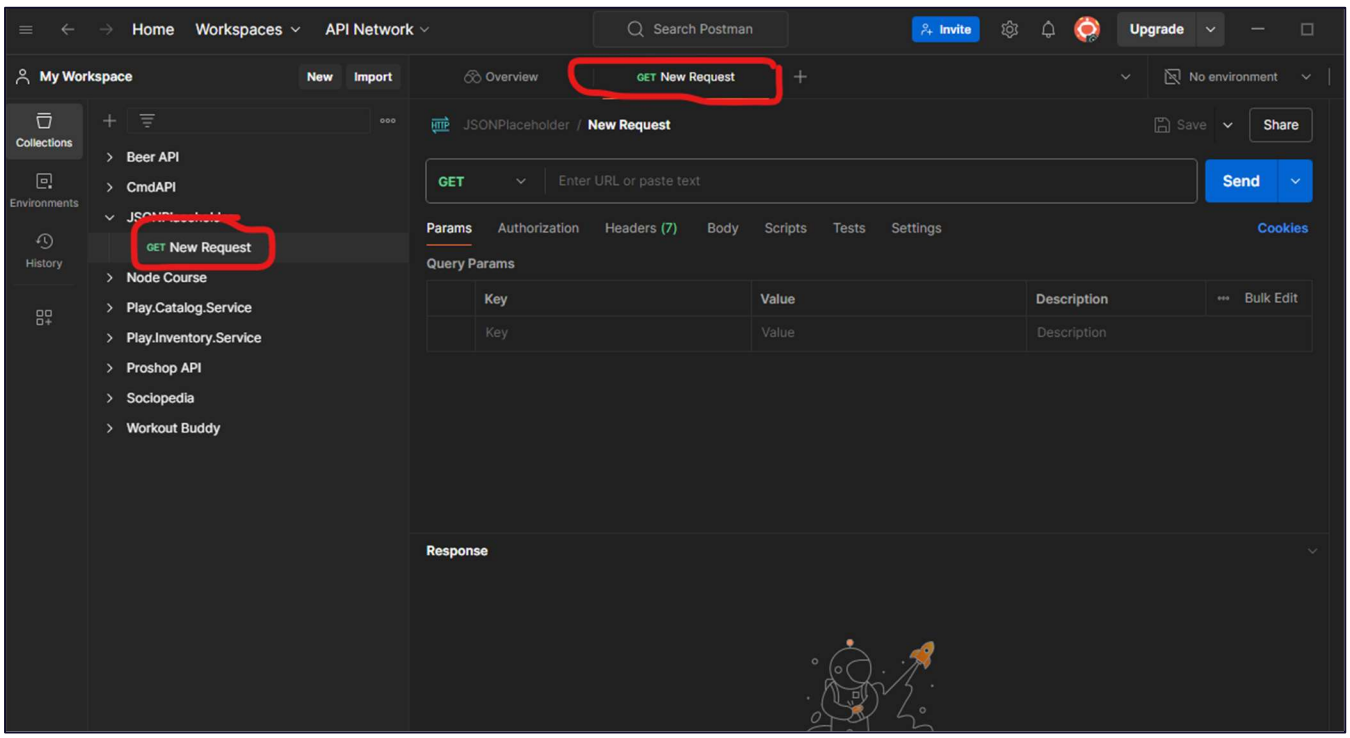
Nu kan vi skapa en Collection för att samla alla typer av anrop som gör mot en och samma REST API. Klicka på plus-ikonen överst i Collections och välj en Blank collection. Nu bör det dyka upp en New Collection i listan under plus-ikonen. Om man för musen ovanför den så dyker det upp en 3-prickar-ikon som man kan klicka på. I den listan som dyker upp så väljer vi Rename och ger vår nya Collection ett bra namn. Till exempel JSONPlaceholder eftersom vi kommer att träna mot den REST API:n. Nu bör det se ut som på figur 4.



Figur 4 Vår nya omdöpta Collection.

En request

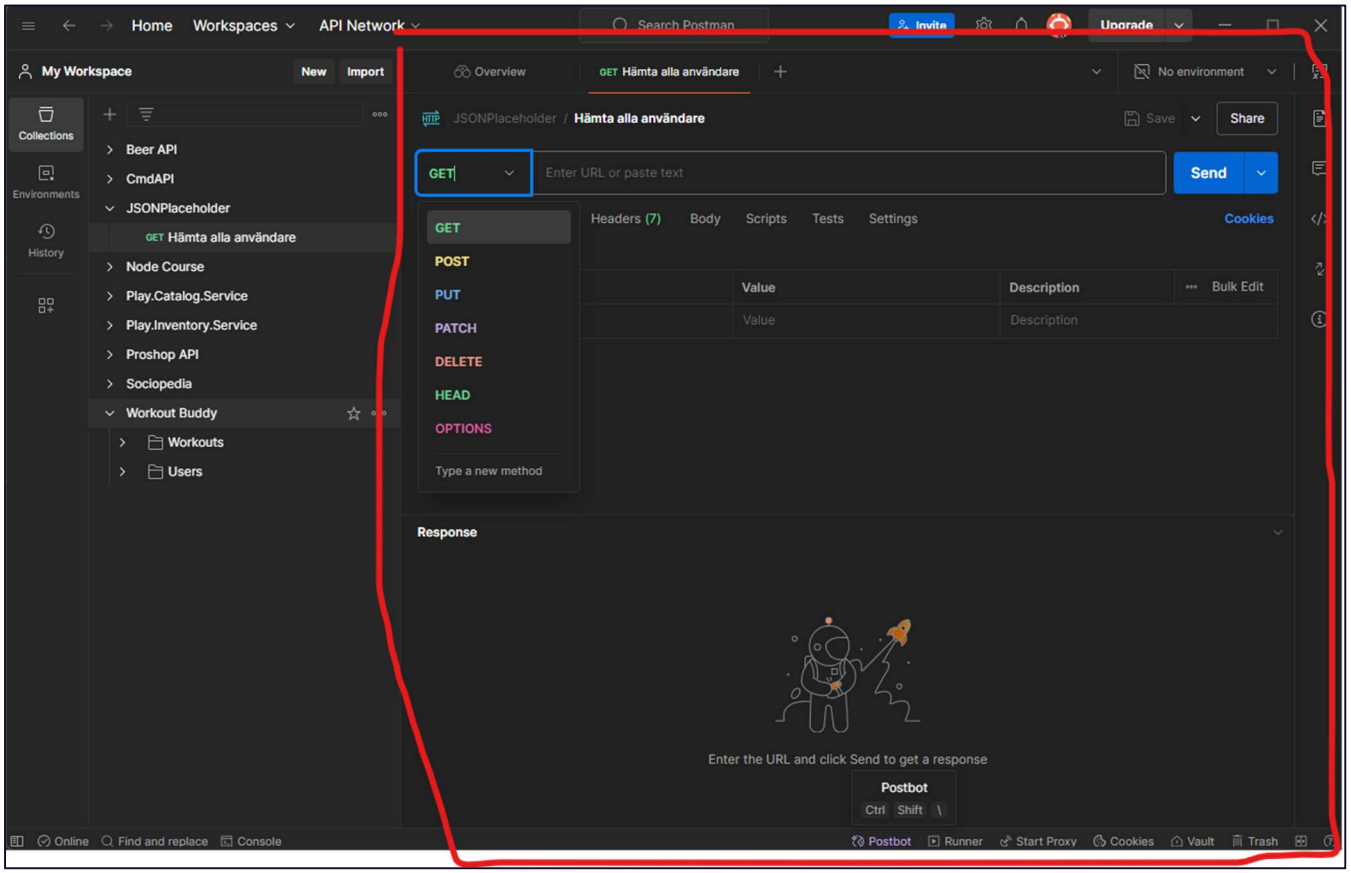
Vi försöker göra vår första request genom att antingen klicka på den blåa länken enligt figur 4. Man kan också föra musen över vår Collection, JSONPlaceholder tills man ser ikonen med 3 prickar och klicka på Add request (ligger som sjätte val uppifrån). Nu skapas det en ny flik på den högra delen av applikationen, se figur 5.



Figur 5 Den nya fliken med vår request.

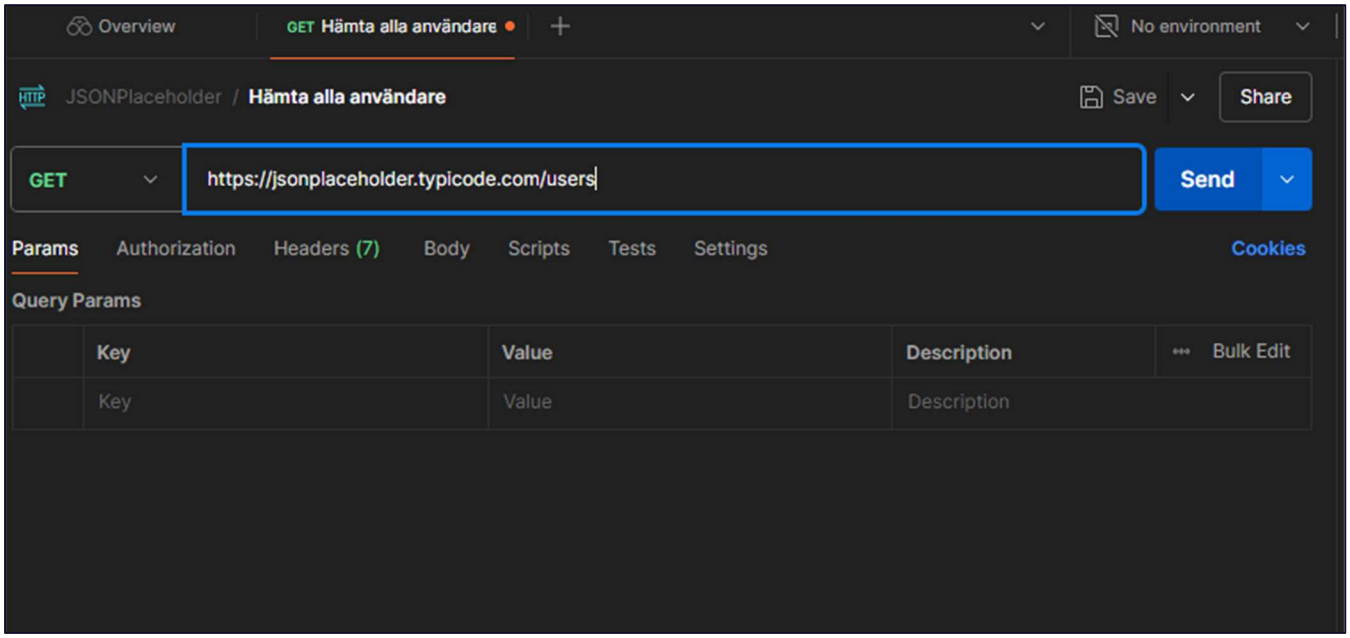
Man kan ändra namn på den här requesten genom att föra musen över New Request till höger. När de 3 prickarna dyker upp klickar man på den och väljer Rename som finns ungefär i mitten. Här kan man skriva in vad det är requesten gör, till exempel Get all users eller Hämta alla användare.

Nu flyttar vi fokus till det stora området till höger, enligt figur 6.



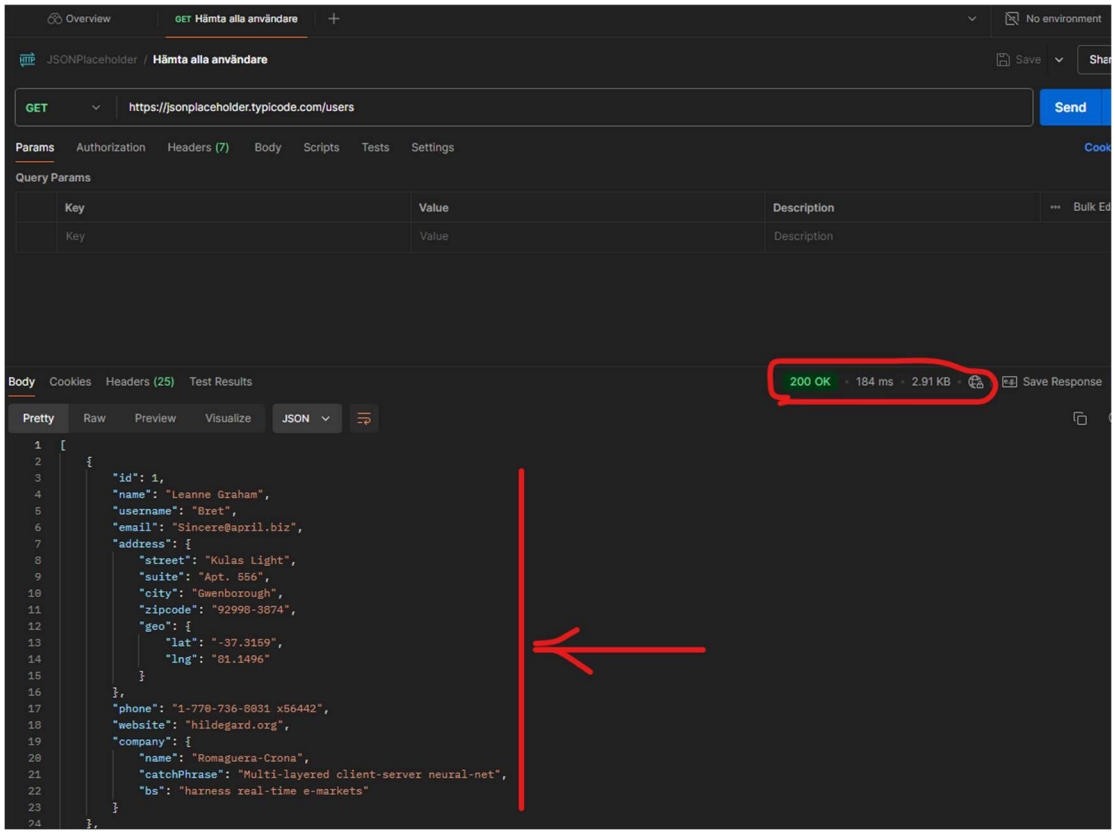
Figur 6 Stora arbetsområdet.

Se till att din nya request-flik är markerad. Under fliken och namnet på din request finns först en dropdown-meny där man kan ställa vilken typ av request man vill göra. Sedan finns det ett långt input-fält där man skriver in adressen till det REST-API man vill anropa. Slutligen på samma rad finns en blå knapp där man skickar iväg sin request när man har fyllt i allting. För den request vi ska göra först borde det se ut som på figur 7.



Figur 7 Typ av request och url för requesten.

Innan vi trycker iväg vår request kan vi trycka på ctrl + S och se till att vår request är sparad. När vi nu trycker på knappen Send så står det Sending request på den nedre delen av det stora området. Man kan dra upp det nedre området och göra det lite större. Men det är här vi ser vilket svar vi får från REST API:et, se figur 8.



Figur 8 Svarsdelen i Postman.

Dels är det intressant att titta uppe till höger för att se att man har fått ett grönt 200 OK. Det innebär att vi har gjort rätt och REST API:et också mår bra. Om man har en annan kod än 200 och det är rött där, då har man gjort fel eller så är REST API:et inte fungerande. Det kan vara olika typer av server-fel eller fel i anropet. Det kan också innebära att man har glömt att infoga en API-nyckel som ibland krävs. Det andra intressanta att titta på är ju själva svaret, om man nu har fått 200 OK. Det finns lite mer till vänster och det är i det här fallet en lista med 10 objekt som har kommit tillbaka från servern.

REST API-dokumentation

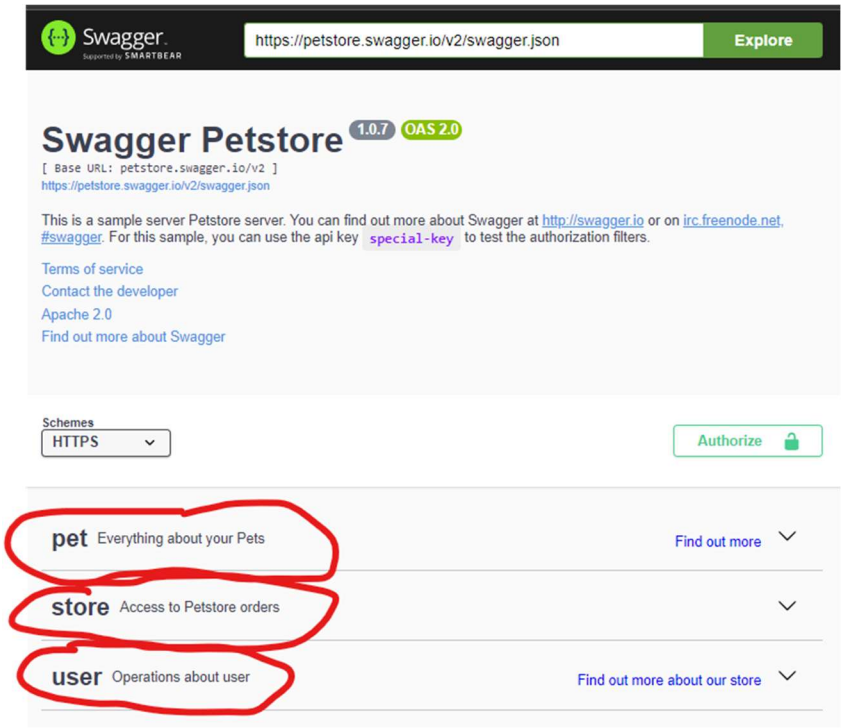
När man skriver ett REST API är det viktigt att man skriver bra dokumentation för användarna. Det kan vara svårt att veta om alla sätt det går att kommunicera med servern om man inte har varit med och utvecklat det själv. För det REST API som vi jobbar med finns det dokumentation på länkarna
https://jsonplaceholder.typicode.com/guide/
https://jsonplaceholder.typicode.com/guide/

Om man till exempel vill hämta alla poster från kan man anropa url:en
https://jsonplaceholder.typicode.com/posts
med en GET request. Det brukar man kalla för en endpoint. Till den endpointen kan man också skicka POST request. Om man vill hämta information om en specifik post kan man anropa url:en https://jsonplaceholder.typicode.com/posts/1 där den sista 1:an i adressen anger vilket id-nummer man är intresserad av. Det är alltså inte samma endpoint som den förra, även fast de är snarlika. Men till den ena skickar man alltså med en extra svans, ett id som för denna REST API är i form av ett tal. Ibland kan ett id vara i form av en sträng också. Det är därför det är viktigt att gå in och läsa dokumentationen för REST API:et så att man vet hur servern förväntar sig att bli anropad.

Swagger

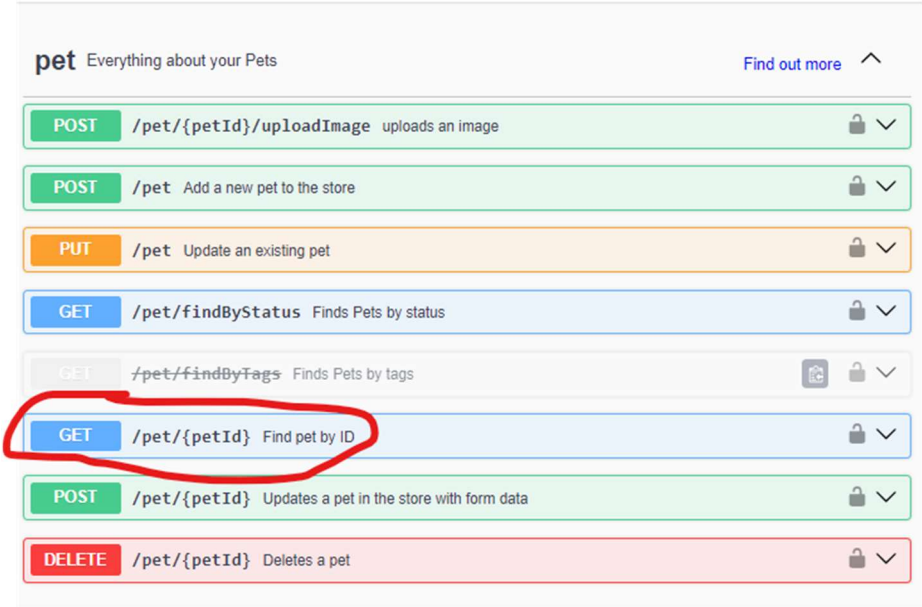
Det finns program för att skriva dokumentation för REST API:er, bland annat ett som heter Swagger. Det är mycket möjligt att man stöter på sådana ibland. Det finns ett exempel på ett sådant på nätet https://petstore.swagger.io/ som alltså är ett så kallat låtsas-API. Fördelen med en dokumentation skriven i Swagger är att man kan testa runt med olika anrop utan att behöva använda Postman. Om vi provar dokumentationen på

petstore.swagger.io så ser man att det är uppdelat på kategorierna **pet**, **store** och **user**, enligt figur 9.



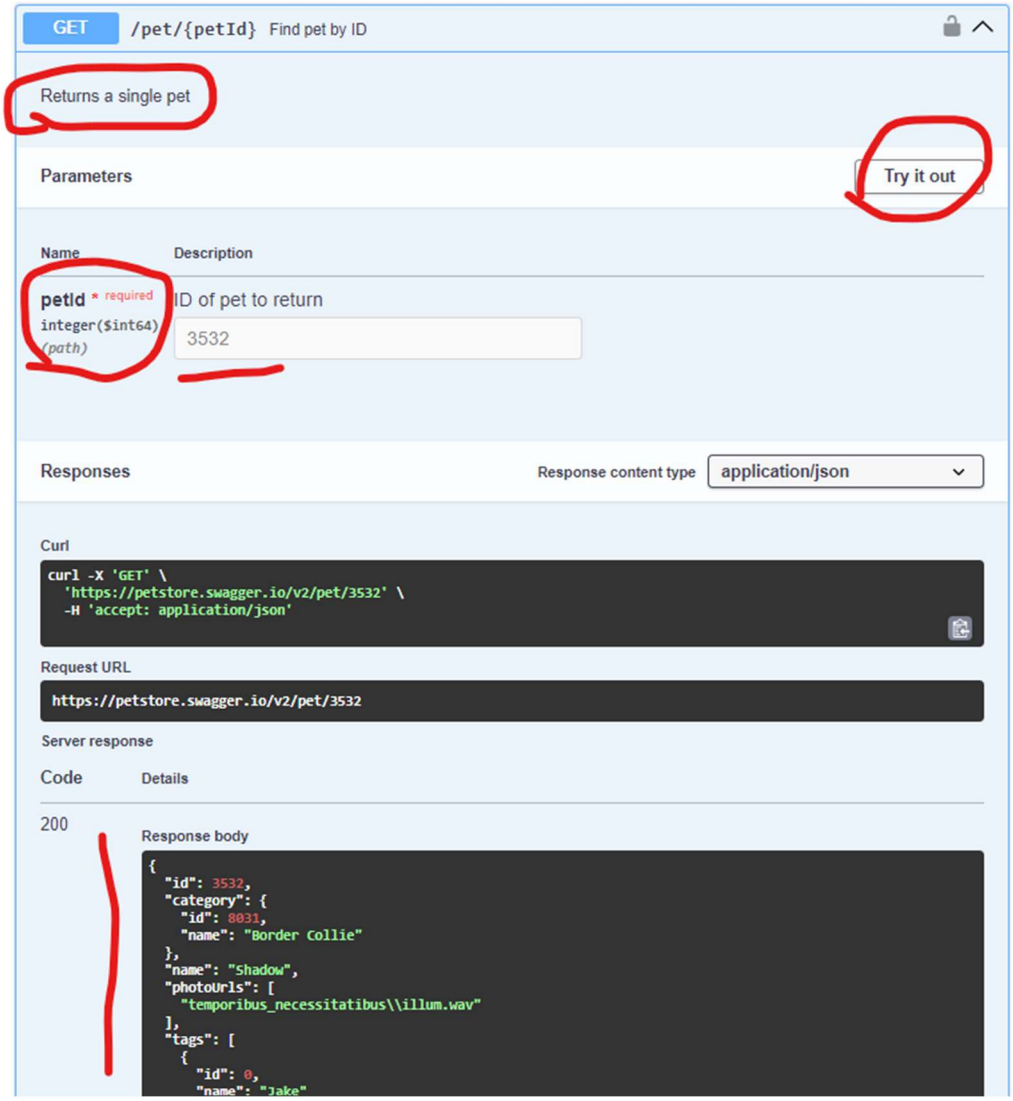
Figur 9 Kategorierna pet, store och user.

Säg att vi vill ta reda på hur det här REST API:et fungerar. Vi kan börja med att kolla på kategorin **pet**, Det finns fyra olika endpoints, men några av dem har olika request-metoder. För endpointen **/pet/{petId}** finns det antingen GET, POST eller DELETE, och för endpointen **/pet** finns det antingen POST eller PUT. Vi ska försöka hitta lite information om ett husdjur, så då vill vi såklart använda GET med endpointen **/pet/{petId}**, enligt figur 10.



Figur 10 Endpoint /pet/{petId}

Det ser lite konstigt ut med petId skrivet innanför klammerparenteser. Tanken är att man ska byta ut det mot ett nummer. Om man klickar på raden som är markerad i figur 10 så finns det några delar som är bra att reda ut, se figur 11.



Figur 11 Förklaring av endpoint-information.

Överst för all information finns det en kort beskrivning av vad den här endpointen skickar tillbaka. Den returnerar alltså ett endaste husdjur. Till höger finns det en inringad knapp som heter **Try it out**. Den kan man klicka på för att möjliggöra testläge av endpointen. När man klickar på den dyker det upp två knappar till. En som heter **Execute** och en som heter **Clear**. Strax ovanför knappen **Execute** finns det en förklaring av den parameter som vi behöver skicka med, alltså **{petId}**. Det står att den ska vara med i form av ett heltal (integer). Så förklaringen till det något kryptiska **/pet/{petId}** är att man ska skriva **/pet/3532** om man vill veta mer om husdjuret med petId 3532. Vill man veta lite mer om husdjur med petId 24 skriver man **/pet/24**. Nu finns inte alla petId-nummer i REST-API:et. Men vi vill kolla upp information om något husdjur.

här manualen används desktop-applikationen. Allting borde se exakt lika ut men det är

POST request

här manualen används desktop-applikationen. Allting borde se exakt lika ut men det är

```
<script src="main.js" type="module" defer></script>
```

Bild 1 Huvudfilen på en webbsidas html-fil.

Lägg noga märke till att attributet type är tillagt. Det ska ha värdet ”module” för att man ska kunna använda sig av moduler.

Export och import

För att kunna dela upp koden så behöver man exportera funktioner, variabler och andra värden till andra filer. Man har bestämt sig för att jobba huvudsakligen i main.js och hämtar in det man behöver till den filen. Det innebär att man behöver importera de delarna till main.js. Man kan antigen använda sig av ”named exports” eller ”default