

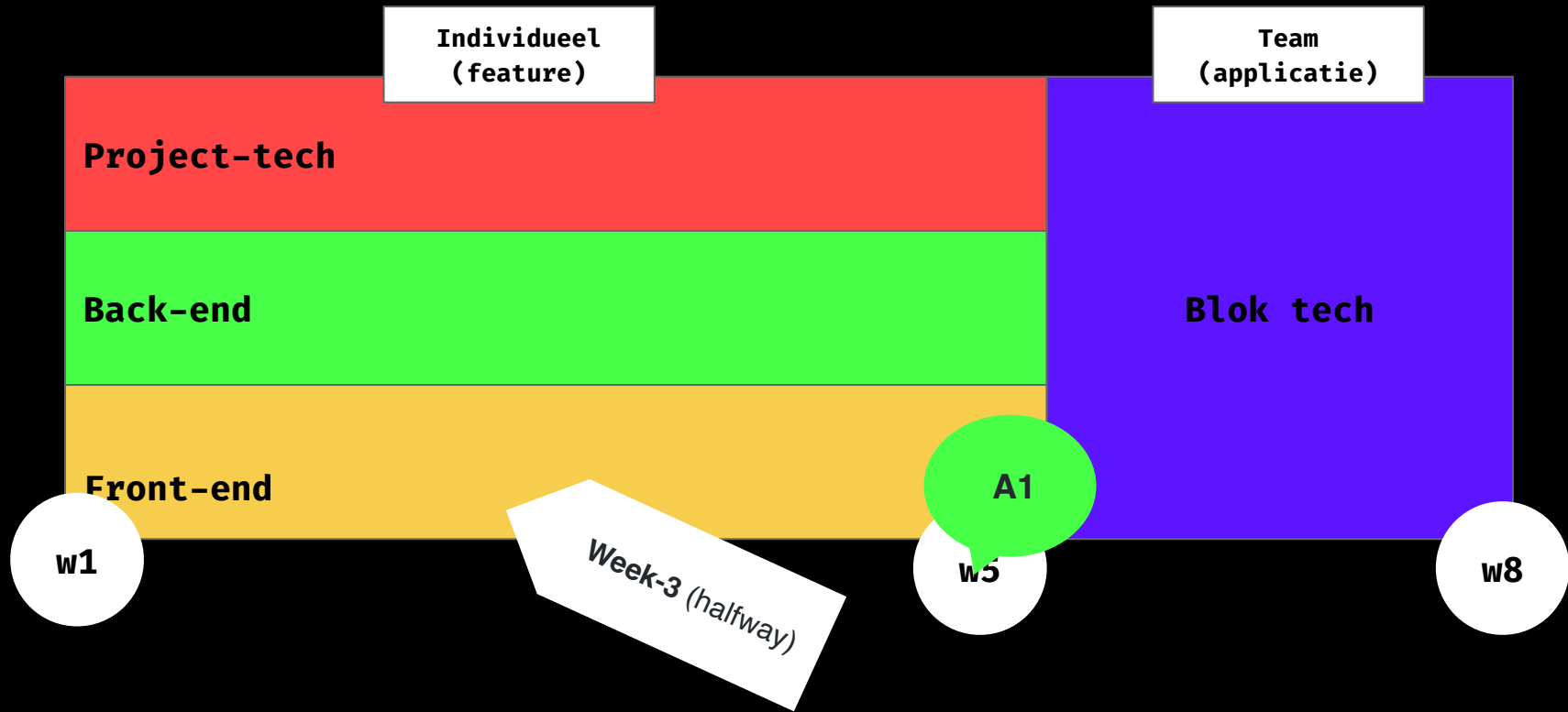
back-end

HTTP & Forms

lab 3/8

Stand-up!

Show what
you did



today

I. ~~Stand-up~~

II. HTTP

III. Forms (+ files)

IV. Connect



HTTP

http

?

The Hypertext Transfer Protocol (HTTP) is an application protocol for distributed, collaborative, and hypermedia information systems. [...]

Hypertext is structured text that uses logical links (hyperlinks) between nodes containing text.

[wikipedia.org](https://en.wikipedia.org)

http

url

A Uniform Resource Locator (URL) [...] is a reference to a web resource that specifies its location on a computer network and a mechanism for retrieving it.

wikipedia.org

http

url

protocol

subdomain

domain

port

path

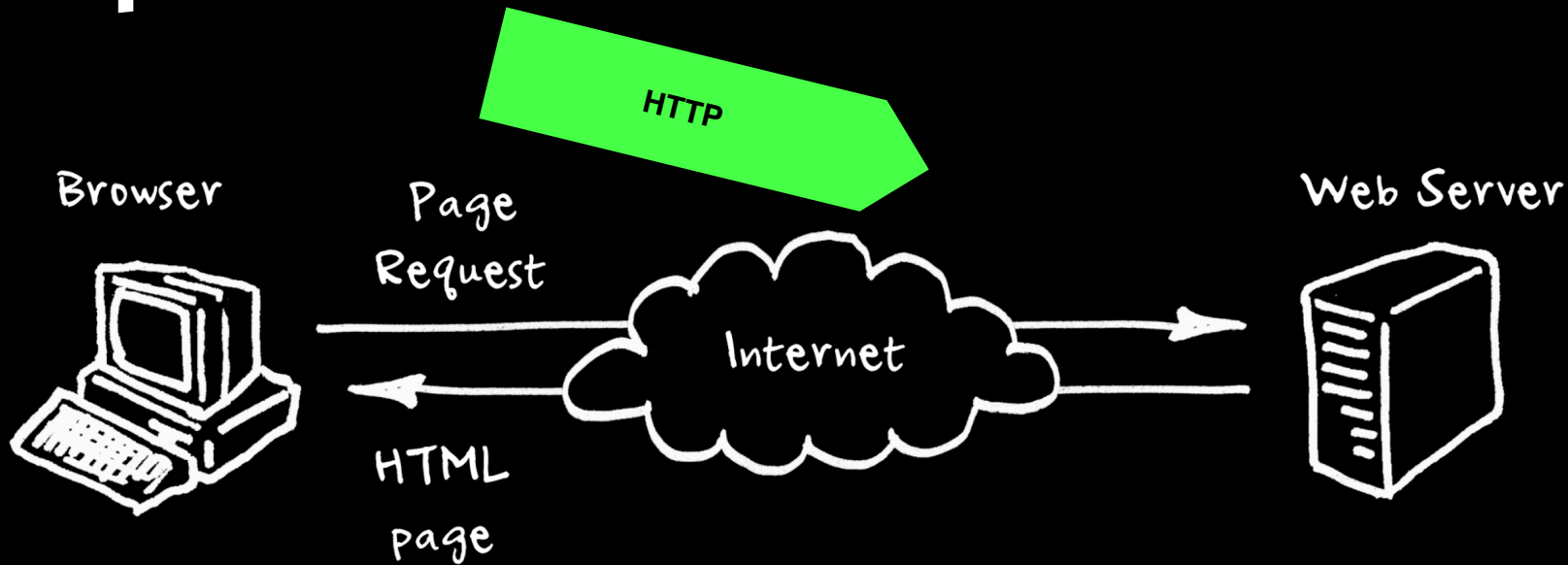
query

fragment

<http://test.example.com:3000/users/search?q=test&w=all#results>

http

req/res



http

response

HTTP/1.1



**status code & status
message**

Date: Mon, 19 Feb 2018 15:40:02 GMT

Last-Modified: Tue, 13 Feb 2018 20:18:22 GMT

Content-Length: 29769

Content-Type: text/html

<!DOCTYPE html... (here comes the 29769 bytes of the requested web page)

http

status

| Category | Range | Example |
|----------------------------------|-------|---------------------------|
| ❖ Information | 1.. | 101 Switching Protocols |
| ❖ Success
Created | 2.. | 200 OK, 201 |
| ❖ Redirect
Permanently | 3.. | 301 Moved |
| ❖ Client error
Found | 4.. | 400 Bad Request, 404 Not |
| ❖ Server Error | 5.. | 500 Internal Server Error |

http

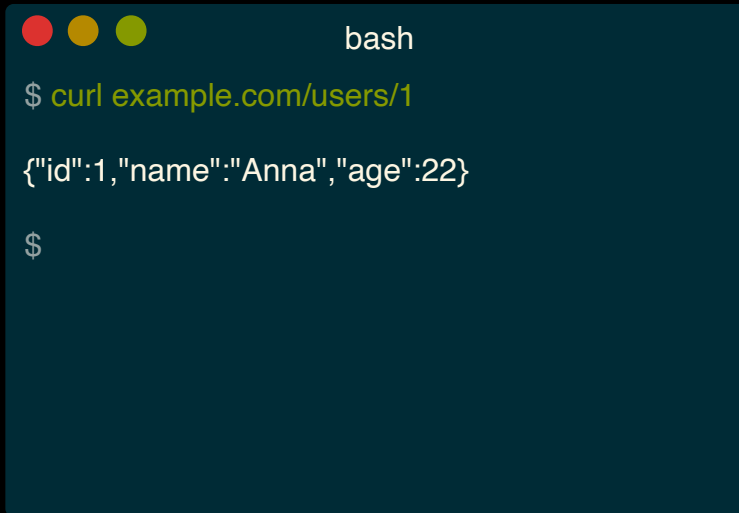
methods

- ❖ **Create:** PUT, POST
- ❖ **Read:** GET
- ❖ **Update:** PATCH
- ❖ **Delete:** DELETE

http

get

```
> /users/1 HTTP/1.1
> Host: example.com
>
< HTTP/1.1 200 OK
<
< {"id":1,"name":"Anna","age":22}
```

A terminal window with a dark blue background and three colored window control buttons (red, yellow, green) in the top-left corner. The prompt is 'bash'. The command '\$ curl example.com/users/1' is entered in yellow text. The output is a JSON object '{"id":1,"name":"Anna","age":22}' displayed in white text. A new prompt '\$' is visible below the output.

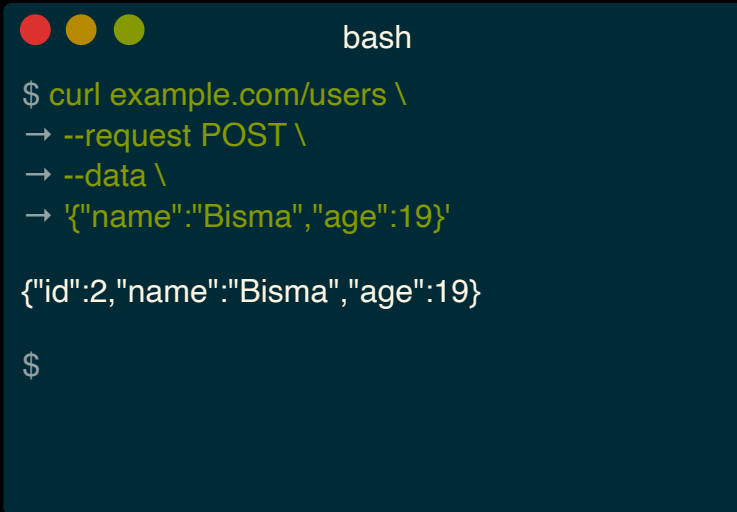
```
bash
$ curl example.com/users/1
{"id":1,"name":"Anna","age":22}
$
```

Request a resource

http

post

```
> /users HTTP/1.1
> Host: example.com
>
> {"name":"Bisma","age":19}
>
< HTTP/1.1 201 Created
< Location: /users/2
<
< {"id":2,"name":"Bisma","age":19}
```

A terminal window with a dark blue background and three colored window control buttons (red, yellow, green) in the top left corner. The prompt is 'bash'. The user enters a curl command to perform a POST request to 'example.com/users' with a JSON body. The output shows a 201 Created status and a Location header pointing to '/users/2', followed by the JSON response.

```
bash
$ curl example.com/users \
  → --request POST \
  → --data \
  → '{"name":"Bisma","age":19}'

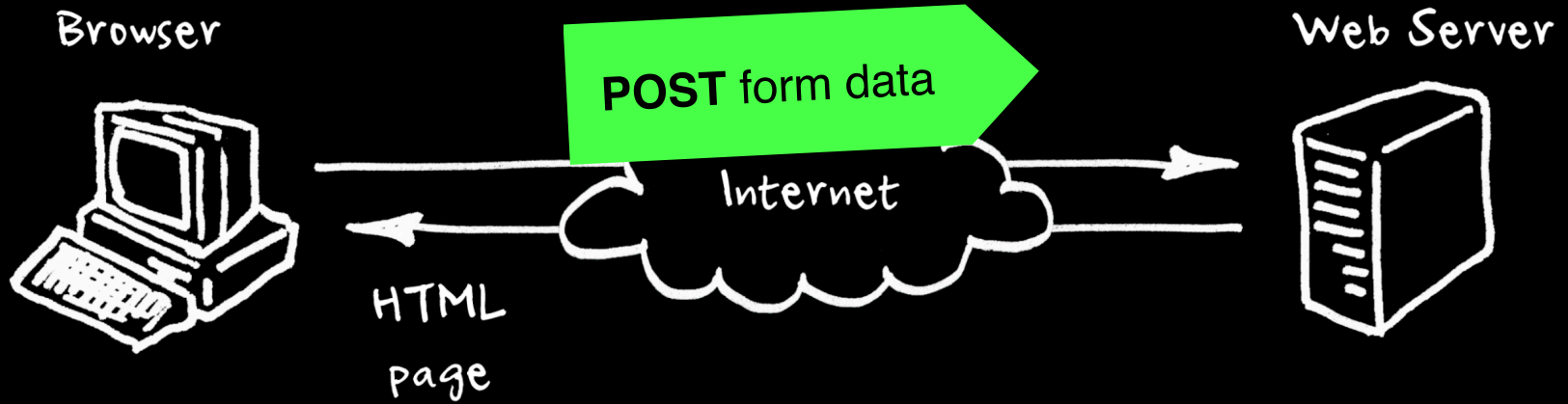
{"id":2,"name":"Bisma","age":19}

$
```

Submit a resource

http

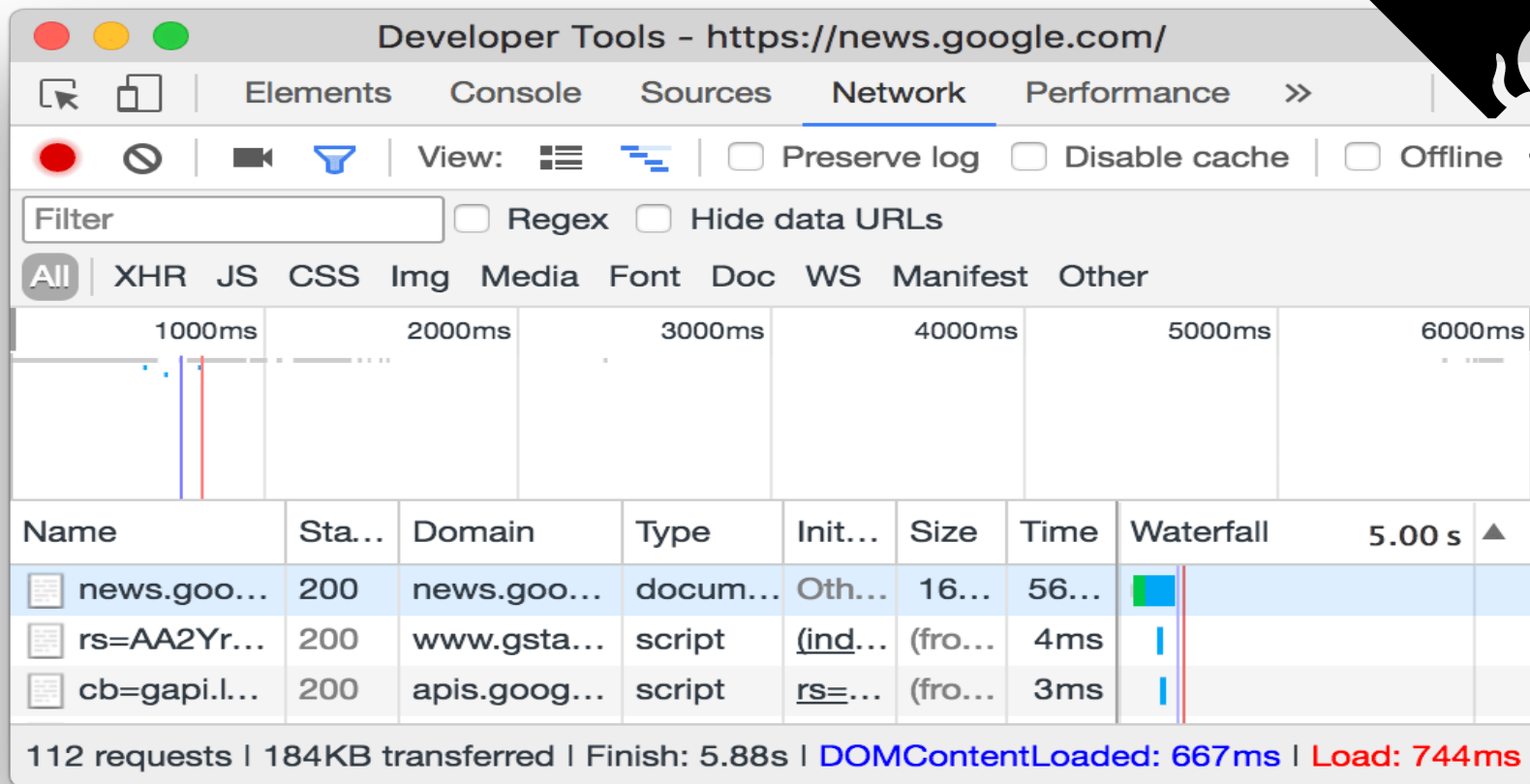
get/post



http

post

- ❖ **Text** text/plain, text/html, text/javascript, ...
- ❖ **Image** image/jpeg, image/png, ...
- ❖ **Audio** audio/webm, audio/ogg, ...
- ❖ **Video** video/webm, video/ogg, ...
- ❖ **Application** application/pdf, application/octet-stream,...



Live demo **network tab**

Forms

localhost:8000/add

Add a new movie

Title

Plot

Description

add

Add a movie

Express

view/add.ejs

```
<% include head.ejs %>
<title>Add a movie - My movie website</title>
<h1>Add a new movie</h1>
<form action=/ method=post>
  <label>Title <input name=title></label>
  <label>
    Plot (short)
    <input name=plot>
  </label>
  <label>
    Description (long)
    <textarea
      name=description
      rows=5
    ></textarea>
  </label>
  <button>Add</button>
</form>
```

Send a **POST** request...

Express

form

slug

// Files

express-server/

├─ node_modules/

├─ static/

│ └─ index.css

├─ view/

│ └─ add.ejs

│ └─ detail.ejs

│ └─ head.ejs

│ └─ list.ejs

│ └─ not-found.ejs

└─ index.js

└─ package.json

```
bash
$ npm install slug body-parser

+ body-parser@1.17.2
+ slug@0.9.1
added 2 packages from 1 contributor and audited 1 package in 2.214s
$
```

slug makes a string (such as a title)
URL safe.

Express

form

body

// Files

express-server/

├─ node_modules/

├─ static/

│ └─ index.css

├─ view/

│ └─ add.ejs

│ └─ detail.ejs

│ └─ head.ejs

│ └─ list.ejs

│ └─ not-found.ejs

└─ index.js

└─ package.json



```
bash
$ npm install slug body-parser

+ body-parser@1.18.2
+ slug@0.9.1
added 2 packages in 1.214s

$
```

body-parser parses most things you give it (json, forms, etc)

Express

index.js

```
var express = require('express')
var find = require('array-find')
var slug = require('slug')
var bodyParser = require('body-parser')

...

express()
  .use(express.static('static'))
  .use(bodyParser.urlencoded({extended: true}))
  .set('view engine', 'pug')
  .set('views', 'views')
  .get('/', movies)
  .post('/', add)
  .get('/add', form)
  .get('/:id', movie)
  .use(notFound)
  .listen(8000)

...
```

Handle a post request to /

Express

index.js

...

```
function form(req, res) {  
  res.render('add.ejs')  
}
```

```
function add(req, res) {  
  var id = slug(req.body.title).toLowerCase()
```

```
  data.push({  
    id: id,  
    title: req.body.title,  
    plot: req.body.plot,  
    description: req.body.description  
  })
```

```
  res.redirect('/') + id  
}
```

...

body-parser parses the data and stores it in **req.body**

Express



Files

files

```
// Files
express-server/
├── node_modules/
├── static/
│   ├── index.css
│   └── index.js
├── view/
│   ├── add.ejs
│   ├── detail.ejs
│   ├── head.ejs
│   ├── list.ejs
│   ├── not-found.ejs
│   └── tail.ejs
├── index.js
└── package.json
```

multer

```
bash
$ npm install multer

+ multer@1.3.0
added 20 packages from 15 contributors
$
```

**multer is middleware for handling
multipart/form-data**

Express

files

folder

```
// Files
express-server/
├── node_modules/
├── static/
│   ├── index.css
│   ├── index.js
│   └── upload/
├── view/
│   ├── add.ejs
│   ├── detail.ejs
│   ├── head.ejs
│   ├── list.ejs
│   ├── not-found.ejs
│   └── tail.ejs
├── index.js
└── package.json
```

We'll upload files to **static/upload**

```
bash
$ npm install multer

+ multer@1.3.0
added 20 packages in 3.041s
```

Express

view/add.ejs

```
<% include head.ejs %>
<title>Add a movie - My movie website</title>
<h1>Add a new movie</h1>
<form
  action=/
  method=post
  enctype=multipart/form-data
>
  <label>Title <input name=title></label>
  <label>
    Cover
    <input name=cover type=file accept=image/*>
  </label>
  <label>
    Plot (short)
    <input name=plot>
  </label>
  ...
  <button>Add</button>
</form>
<% include tail.ejs %>
```

Accept only images

Express

index.ejs

```
...  
var multer = require('multer')  
  
...  
  
var upload = multer({dest: 'static/upload/'})  
  
express()  
  ...  
  .post('/', upload.single('cover'), add)  
  ...  
  
function add(req, res) {  
  ...  
  data.push({  
    ...  
    cover: req.file ? req.file.filename : null,  
    ...  
  })  
}  
  
...
```

multer sets req.file

Express

localhost:8000/add

Add a new movie

Wonder Woman

Diana, an Amazonian warrior...

Cover

wonder-woman.jpg

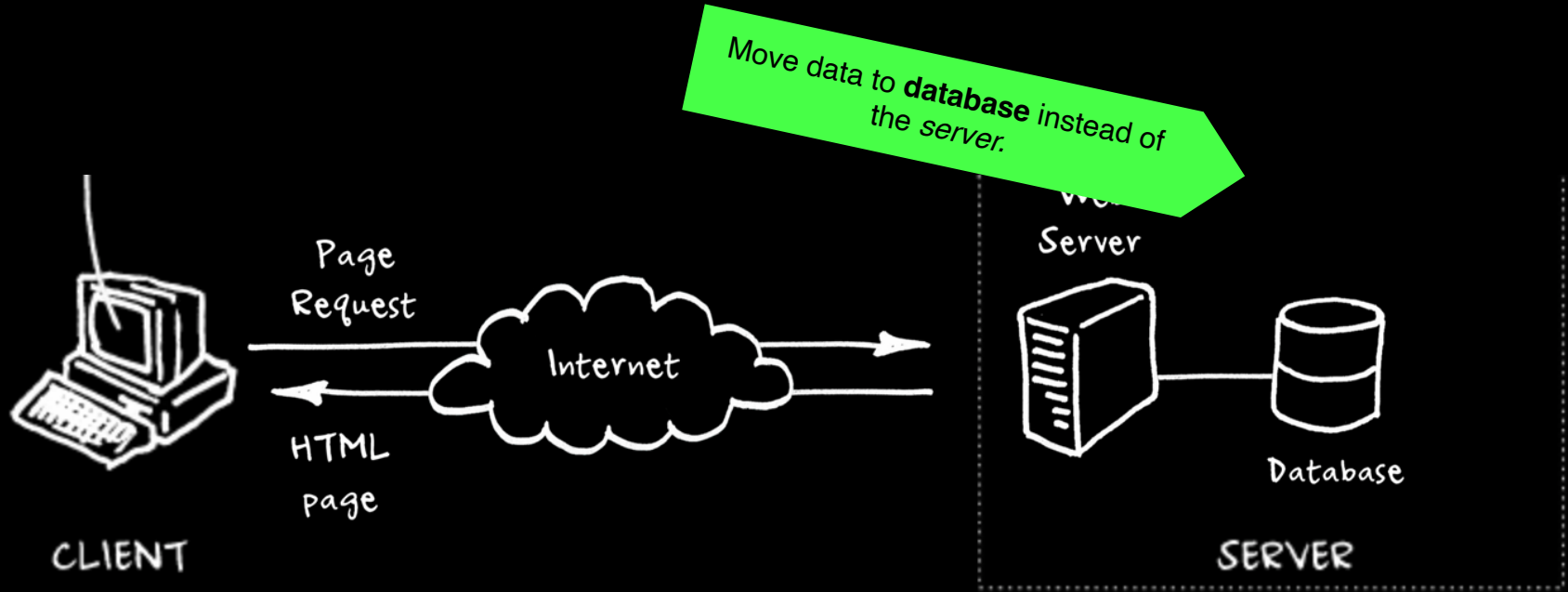
When a pilot crashes and tells of conflict in the outside world, Diana, an Amazonian warrior in training, leaves home ...

We can add files!

Express



Connect



connect

mongodb

MongoDB (from humongous) is a free and open-source cross-platform document-oriented database program. Classified as a NoSQL database program, MongoDB uses JSON-like documents with schemas. MongoDB is developed by MongoDB Inc. [...]

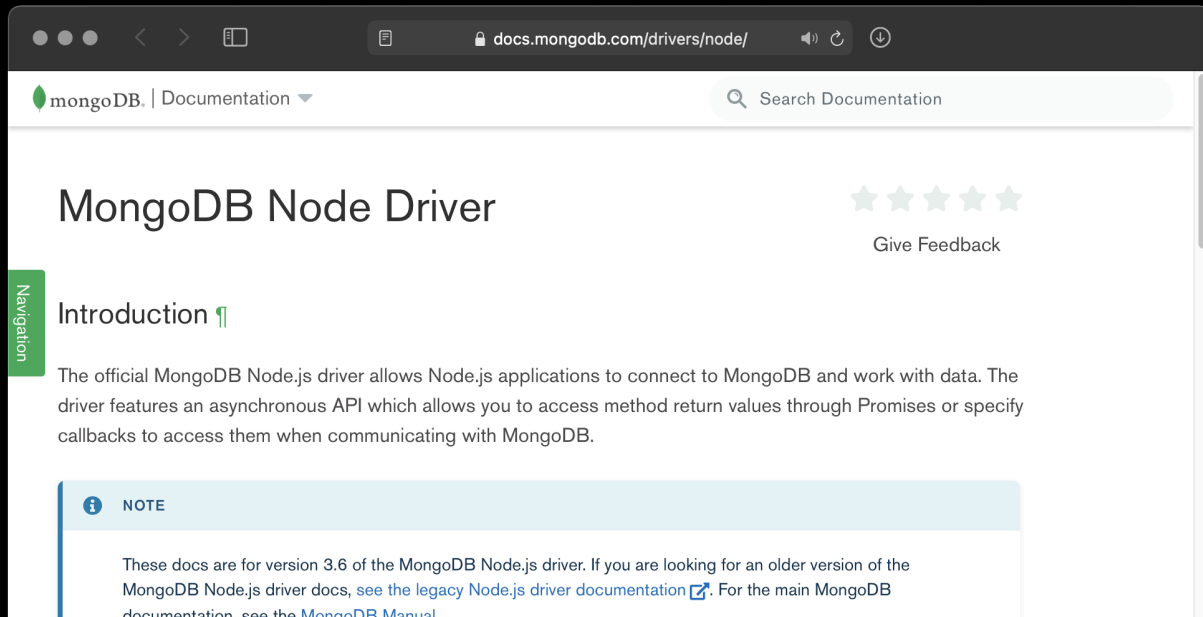
[wikipedia.org](https://www.wikipedia.org)

connect

mongodb

- ❖ JavaScript can be used in queries, aggregation function
- ❖ Map-reduce can be used for batch processing of data and aggregation operations
- ❖ Manage massive increases in new, rapidly changing data types

<https://www.mongodb.com/compare/mongodb-mysql>



Note: there are a lot of small steps involved. Read the Mongo guides very carefully. If you miss a step everything will be broken.



bash

```
$ npm install mongodb dotenv
```

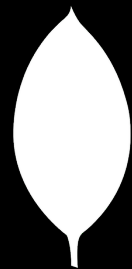
```
+ mongodb@2.2.33
```

```
+ dotenv@4.0.0
```

```
added 11 packages in 4.022s
```

```
$
```

mongodb wraps **MongoDB**
for **Node**

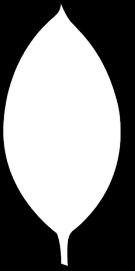


connect

mongodb

```
// Files
mongodb-server/
├── node_modules/
├── static/
│   ├── index.css
│   ├── index.js
│   └── upload/
├── view/
│   ├── add.ejs
│   ├── detail.ejs
│   ├── head.ejs
│   ├── list.ejs
│   ├── not-found.ejs
│   └── tail.ejs
├── index.js
└── package.json
```

```
.env
DB_HOST=localhost
DB_PORT=27017
DB_NAME=mymoviewebsite
DB_USERNAME=dandevri
```



connect

mongodb

```
// Files
mongodb-server/
├── node_modules/
├── static/
│   ├── index.css
│   ├── index.js
│   └── upload/
├── view/
│   ├── add.ejs
│   ├── detail.ejs
│   ├── head.ejs
│   └── list.ejs
```

```
.env
DB_HOST=localhost
DB_PORT=27017
DB_NAME=mymoviewebsite
```

Note: Never ever put your **host and password in code or on GitHub!** People will be able to access your database!

connect

mongodb

```
// Files
mongodb-server/
├── node_modules/
├── static/
│   ├── index.css
│   ├── index.js
│   └── upload/
├── view/
│   ├── add.ejs
│   ├── detail.ejs
│   ├── head.ejs
│   ├── list.ejs
│   ├── not-found.ejs
│   └── tail.ejs
└── .env

├── index.js
└── package.json
```



ii mor

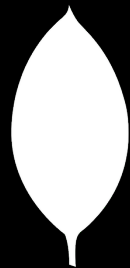
```
index.js
...
var multer = require('multer')
var mongo = require('mongodb')

require('dotenv').config()

var db = null
var url = 'mongodb://' + process.env.DB_HOST + ':' + process.env.DB_PORT

mongo.MongoClient.connect(url, function (err, client) {
  if (err) throw err
  db = client.db(process.env.DB_NAME)
})
...
```

mong
odb



index.js

...

```
function form(req, res) {  
  res.render('add.ejs')  
}
```

```
function add(req, res) {  
  var id = slug(req.body.title).toLowerCase()
```

```
  data.push({  
    id: id,  
    title: req.body.title,  
    plot: req.body.plot,  
    description: req.body.description  
  })
```

```
  res.redirect('/') + id  
}
```

...

title, plot, and description come from **name** attributes on inputs

Express

Wonder Woman

When a pilot crashes and tells of conflict in the outside world, Diana, an Amazonian warrior in training, leaves home to fight a war, discovering her full powers and true destiny.

...and we can submit the form!

localhost:8000/wonder-woman

Wonder Woman

When a pilot crashes and tells of conflict in the outside world, Diana, an Amazonian warrior in training, leaves home to fight a war, discovering her full powers and true destiny.

remove

...but we cannot **remove** movies?
And we can't **update** them.

Express

input



Receive input from users on the server and manipulate that data for your own feature using HTTP request methods.

Synopsis

- **Time:** 10:00h
- **Goals:** subgoal 4, subgoal 5, subgoal 6
- **Due:** before week 4

Description

So far we only send data (response) to the client with our server. A one-sided conversation. Now the fun starts, it's time to actually start receiving data from users. For example; users can enter something into an input field or submit whole forms with file uploads.

The description of this assignment is quite vague since the end result will be very specific to your Job Story. Make sure you at least spend the

work on **input and connect**

input



Receive input from users on the server and manipulate that data for your own feature using HTTP request methods.

Synopsis

- **Time:** 10:00h
- **Goals:** subgoal 4, subgoal 5, subgoal 6
- **Due:** before week 4

Description

Note: *Input* is quite a ‘**large**’ and ‘**vague**’ assignment since the end result will be very specific to your Job Story.



exit;

see you in **lab-4!**