Node.js

Veerle Ongenae

Overzicht

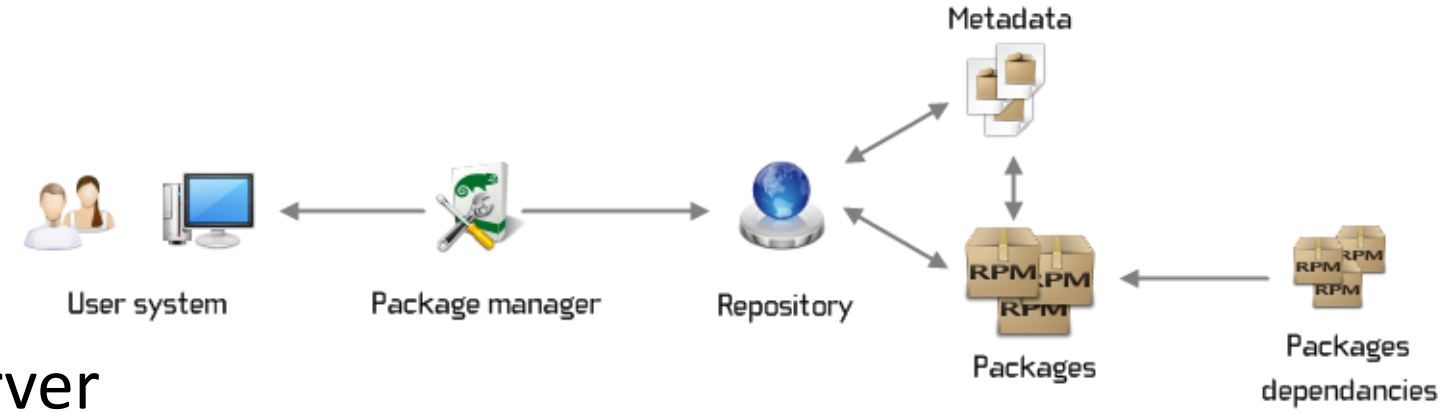
- Node.js en npm

Wat is Node.js?

- Javascript buiten de browser
  - Commandolijn programma's
  - Serverside platform
- Gebouwd op Chromes Javascript Runtime
- https://nodejs.org/en/download

npm

- Node package manager
- Pakketbeheerder voor Node.js
  - Repository: publiceren open-source Node.js projecten (=packages)
  - Beheren van packages
    - Interageren met repository
    - Packages installeren
    - Versiebeheer
    - Beheer afhankelijkheden
- Bv. HTTP-module → webserver
- Deel van de Node.js-installatie
- Alternatief: yarn (https://classic.yarnpkg.com/en/)



https://devopedia.org/package-manager

package.json

- Informatie over het project
- In de hoofdmap

```
{
  "name" : "MyApp",
  "version" : "1.0.0",
  "dependencies" : {
      "sax" : "0.3.x",
      "nano" : "*",
      "request" : ">0.2.0"
  }
}
```

afhankelijkheden: gebruikte packages

```
{
  "name": "book-exampleproject",
  "version": "1.0.0",
  "description": "This is …",
  "main": "server.js",
  "repository": {
    "type": "git",
    "url": "..."
   },
   "dependencies": {
     "express": "latest",
     "mongoose": "latest"
   },
  "author": "Joerg Krause",
  "license": "MIT",
  "homepage": "http://www.joergkrause.de"
}
```

startpunt uitvoeren programma

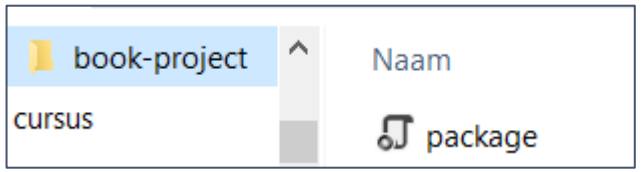
Applicatie maken

```
C:\Users\vongenae\Documents\webtech\Webtechnologies\2017-2018\voorbeelden\11 nodejs>mkdir book-project

C:\Users\vongenae\Documents\webtech\Webtechnologies\2017-2018\voorbeelden\11 nodejs>cd book-project

C:\Users\vongenae\Documents\webtech\Webtechnologies\2017-2018\voorbeelden\11 nodejs\book-project>npm init
```

```
{
  "name": "book-project",
  "version": "1.0.0",
  "description": "Voorbeeld NodeJS",
  "main": "server.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "",
  "license": "ISC"
}
```



nuttige scripts voor het programma

- testen
- build
- …

Applicatie starten

```
npm start
```

main in package.json uitvoeren

```
npm run <task-name>
```

package.json

```
{
  …
  "scripts": {
    "start-dev": "node lib/server-development",
    "start": "node lib/server-production"
  },
  …
}
```

Afhankelijkheden

UNIVERSITEIT GENT

- Opgeven in package.json
  - "express": "~4.8.6“
  - ~ → patch meest recente

major
minor
patch
build

- Installatie via npm
  - Globaal (alle projecten)

```
npm install <naampakket> -g
```

--global

  - Lokaal (één project)

```
npm install <naampakket> --save
```

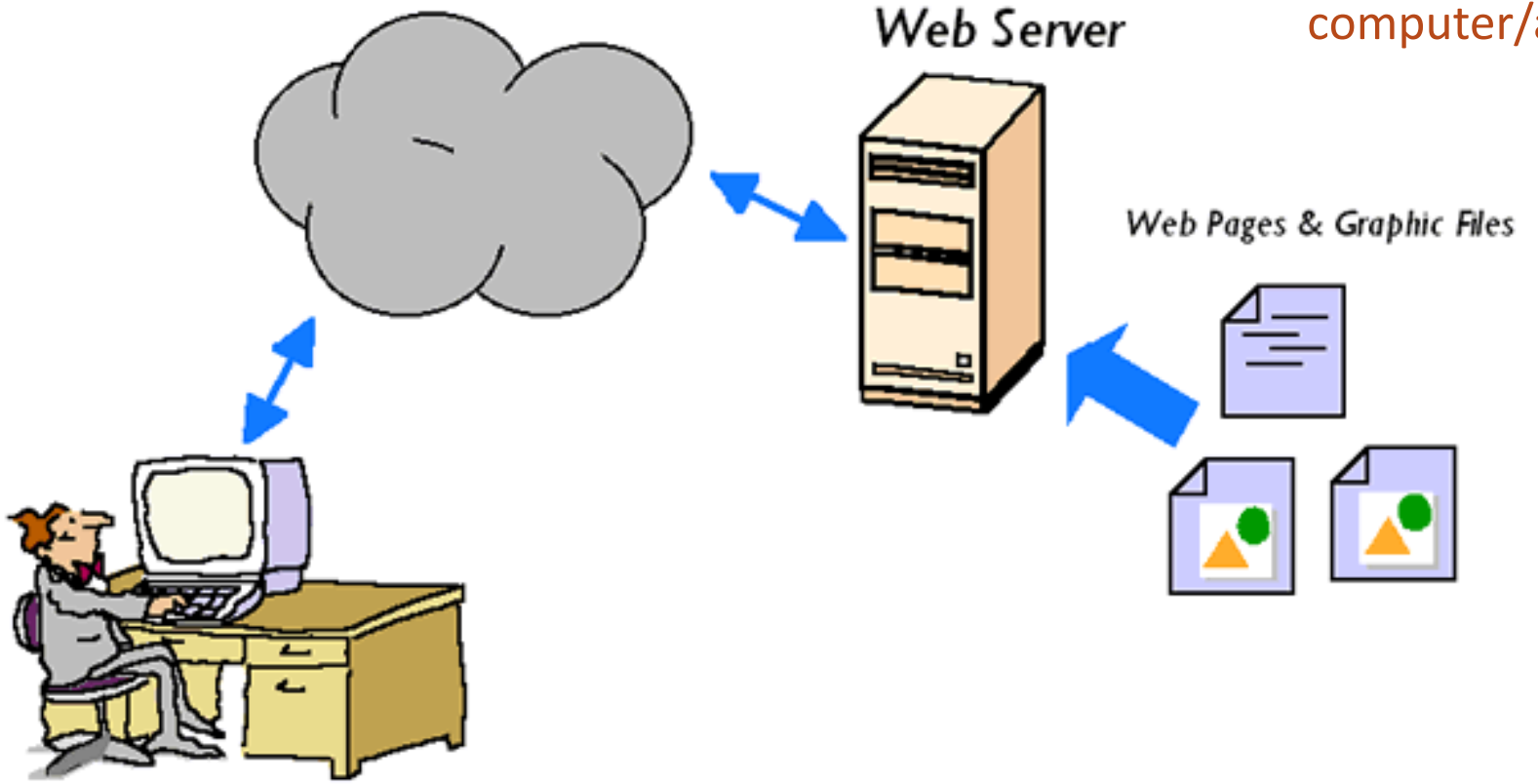
toegevoegd in package.json

➢ In map node_modules

Overzicht

- Node.js en npm
- HTTP-server

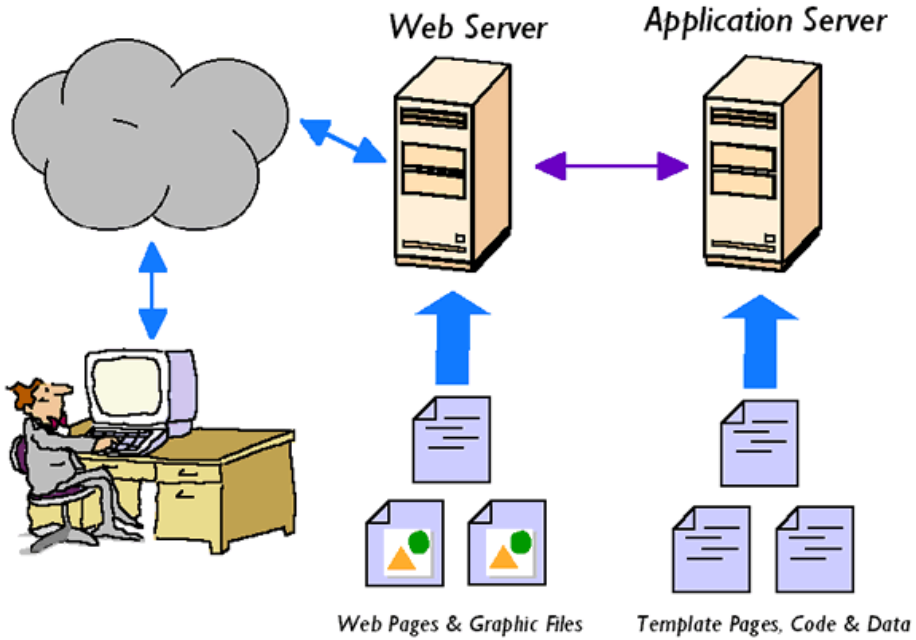
Webservers: statische pagina's



bron: http://www.resultantsys.com/index.php/general/what-is-a-web-application-server/

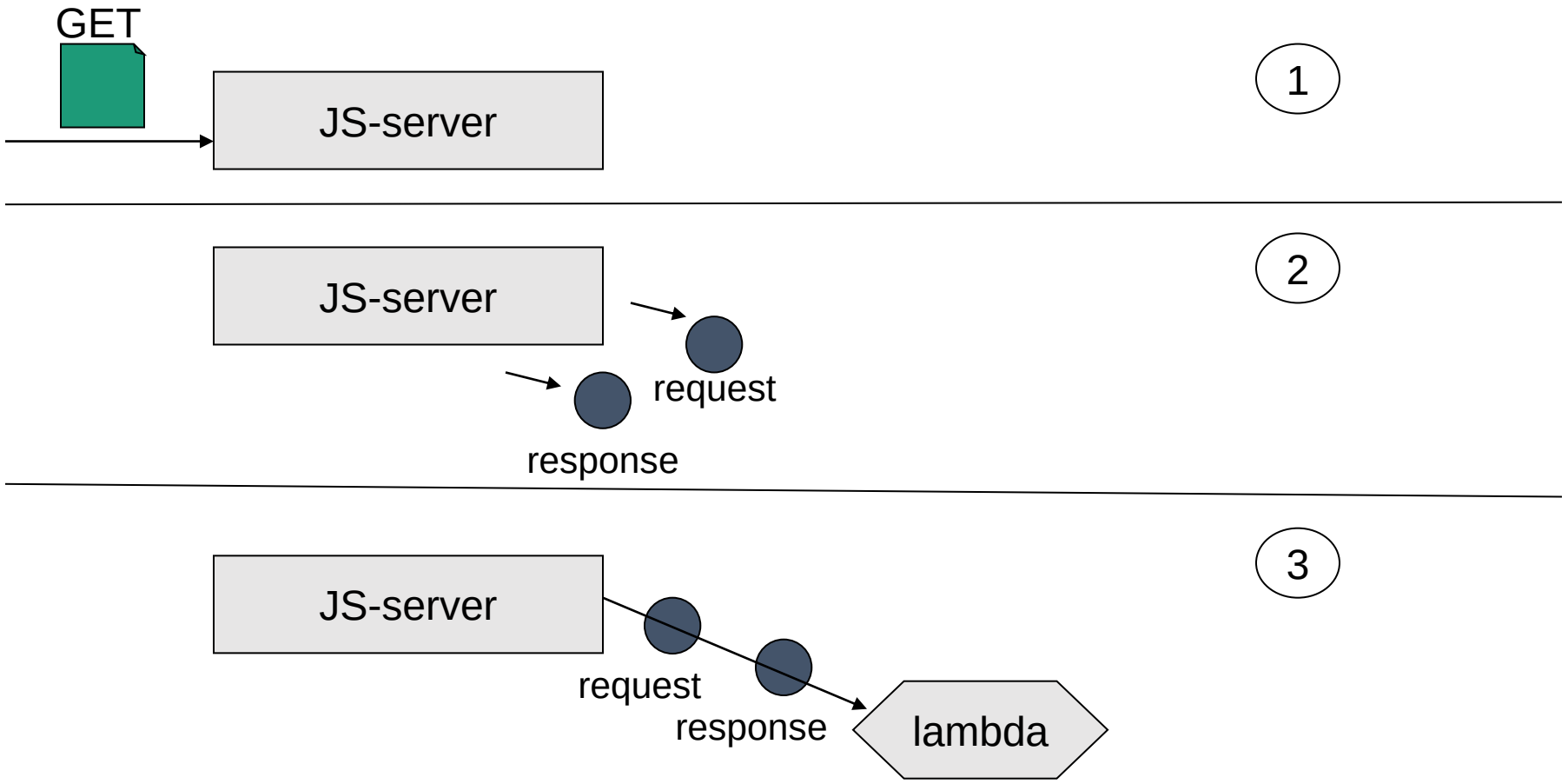
Dynamische webpagina's

- Webpagina's genereren
  - Op server
  - Door "programma's"
    - Geïnterpreteerd (PHP, node.JS, …)
    - Gecompileerd (Java Spring, ASP.NET core, …)

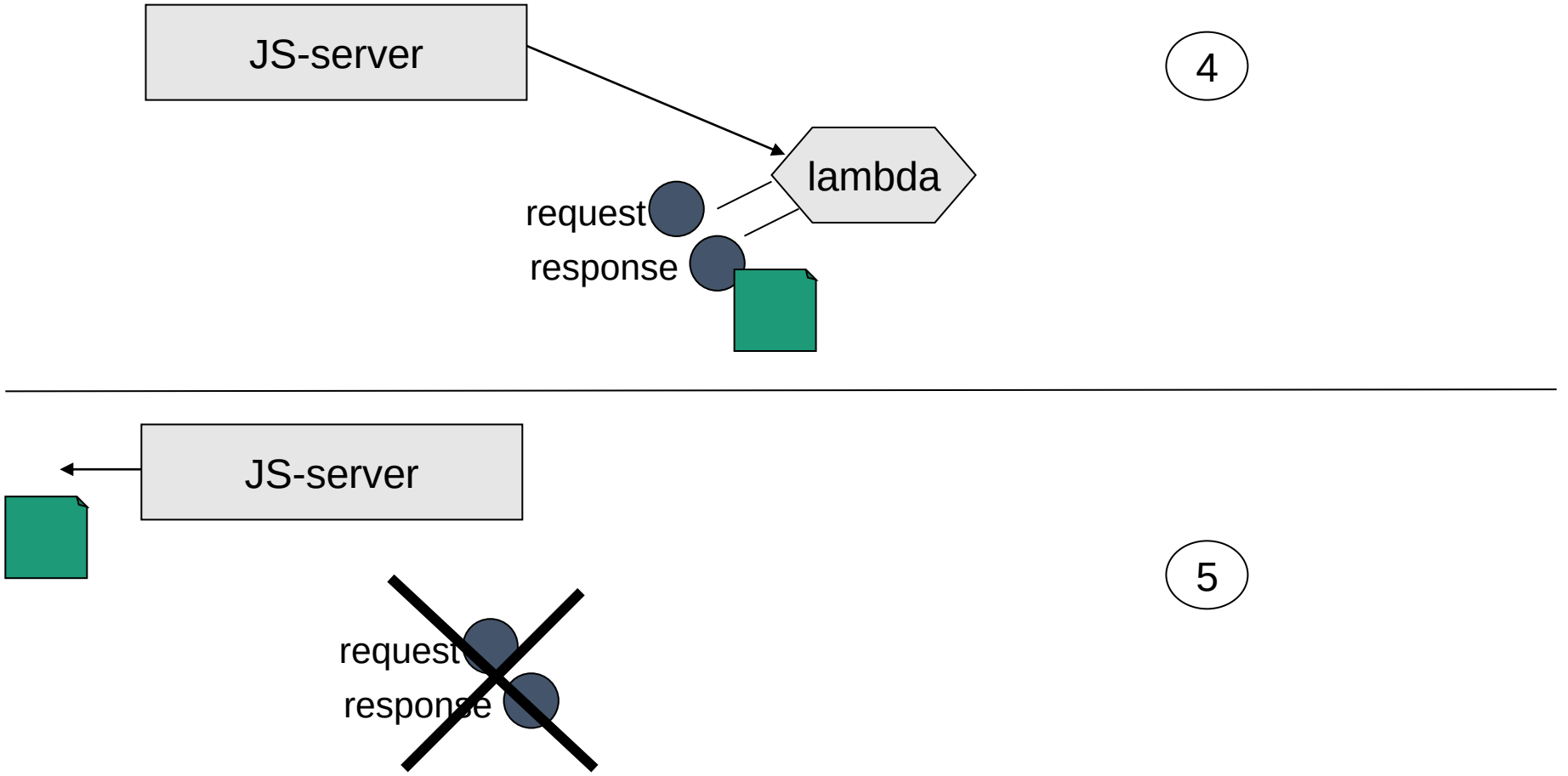


bron: http://www.resultantsys.com/index.php/general/what-is-a-web-application-server/

HTTP-server


GET
JS-server
1
JS-server
2
request
response
JS-server
3
request
response
lambda

HTTP-server


JS-server
4
lambda
request
response
JS-server
5
request
response

HTTP-server in Node.js

```
let http = require('http');
let fs = require('fs');
let port = process.env.port || 1337;

http.createServer(
  (req, res) => {
    console.log("Aanvraag op poort 1337");
    res.writeHead(200,
    {
      'Content-Type': 'text/html',
      'Access-Control-Allow-Origin': '*'
    });
    let read = fs.createReadStream(__dirname + '/index.html');
    read.pipe(res);
  }).listen(port);
```

modules, https kan ook

poort

HTTP-server maken op poort

request en response-object

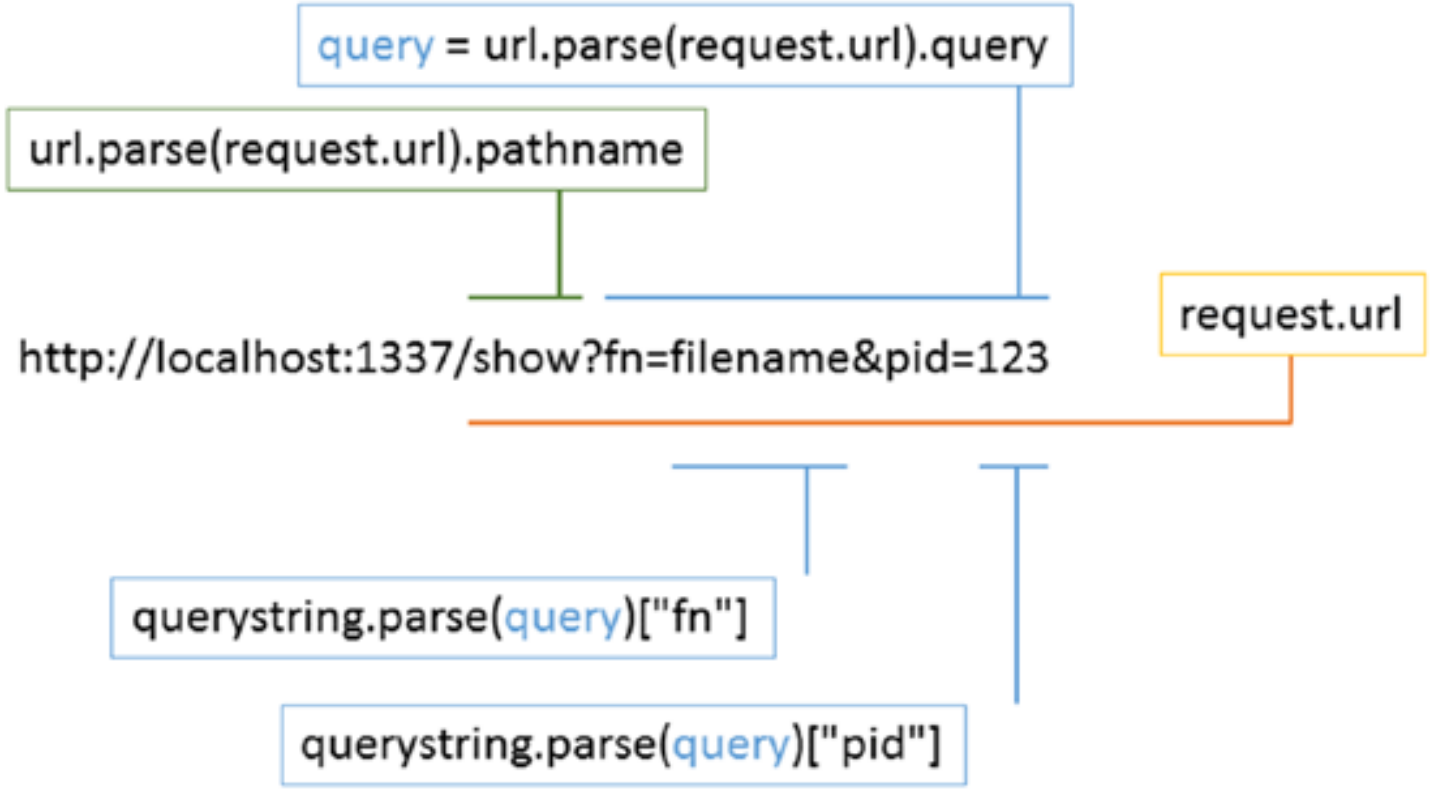
headers instellen

afhandelen aanvraag

map huidige module/app bestand inlezen en wegschrijven naar body response

Stukken URL opvragen

module url



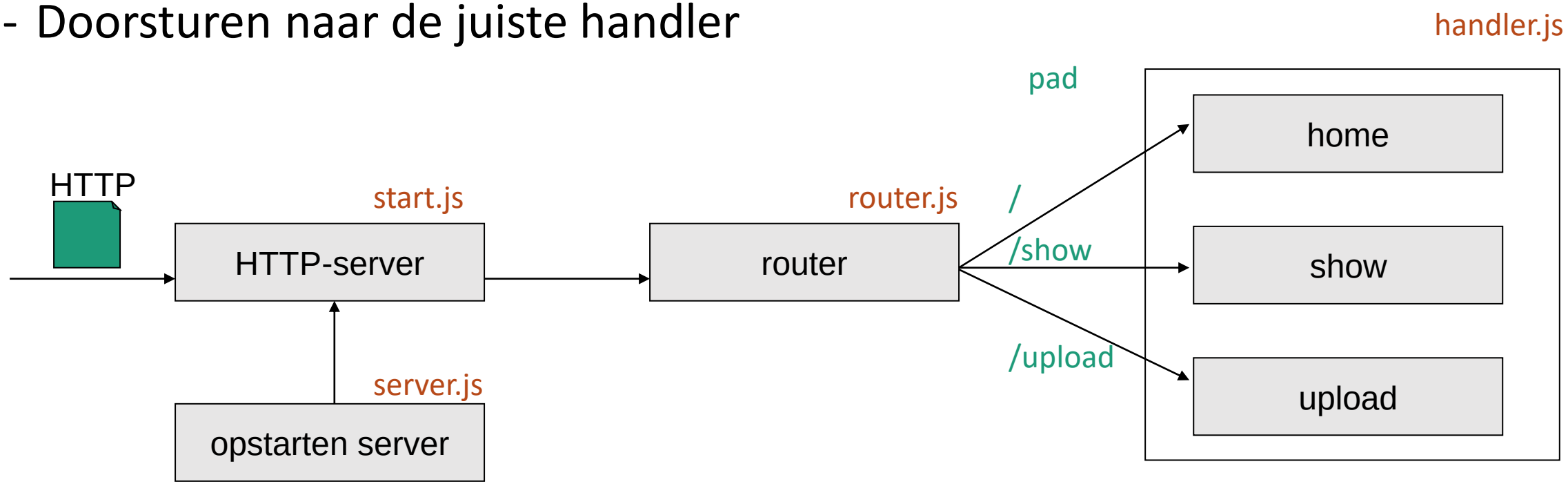
module querystring

Overzicht

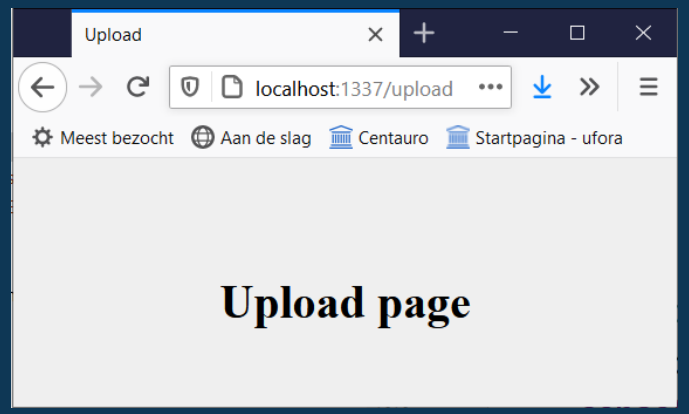
- Node.js en npm
- HTTP-server
- Routing

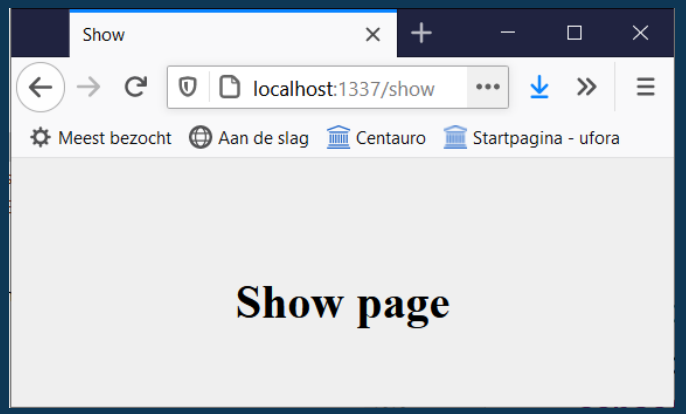
HTTP-server: routing

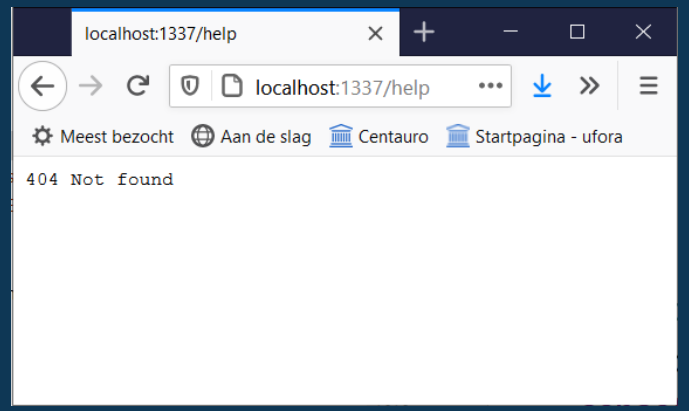
- Meerdere pagina's
- Doorsturen naar de juiste handler

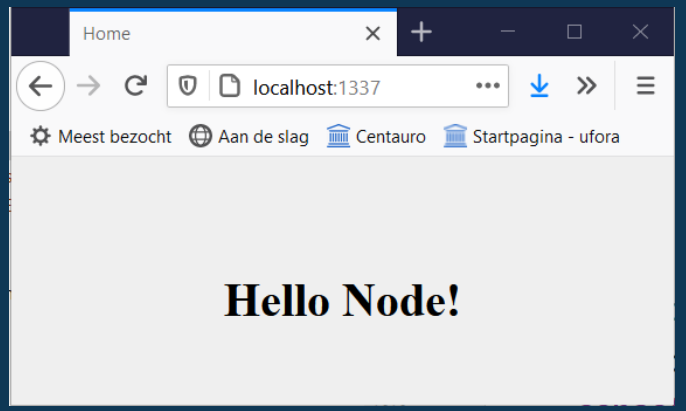


```
Starting.
Has been started.
Request for /upload
Request for /show
Request for /help
No Method found for /help
Request for /
```









Voorbeeld routing

Voorbeeld routing – opstarten server

server.js

```
let server = require('./start');
let router = require('./router');
let requestHandlers = require("./handlers");

let handler = {};
handler["/"] = requestHandlers.home;
handler["/show"] = requestHandlers.show;
handler["/upload"] = requestHandlers.upload;

server.start(router.route, handler);
```

zelfgeschreven modules

link pad ~ functie

server starten

router en handlers meegeven

Voorbeeld – handler-functies

handlers.js

```
let fs = require('fs');
function home(response) {
  respond(response,'views/home.html’);
  return true;
}
function show(response) {…}
function upload(response) {…}
function respond(response, file) {
  fs.readFile(file, (err, data) => {
    response.writeHead(200, {"Content-Type": "text/html"});
    response.write(data);
    response.end();
  });
}
exports.home = home;
exports.show = show;
exports.upload = upload;
```

module filesystem

handler-functies

kopieert file naar body HTTP-bericht

functies exporteren

Voorbeeld - router

UNIVERSITEIT GENT

router.js

```
function route(pathname, handler, response) {
  console.log("Request for " + pathname);
  if (handler[pathname] !== undefined) {
    return handler[pathname](response);
  } else {
    console.log("No Method found for " + pathname);
    return null;
  }
}
exports.route = route;
```

bepaalt de juiste handler-functie voor het gegeven pad en voert die uit

functie exporteren

Voorbeeld - server

start.js

modules

server opstarten

afhandelen aanvraag

pad bepalen

inhoud voor specifiek pad

headers bij fout instellen

body bij fout uitschrijven

antwoord doorsturen

HTTP-server aanmaken

functie exporteren

```
let http = require("http");
let url = require("url");

function start(route, handler) {
  console.log("Starting.");

  function onRequest(request, response) {
    let pathname = url.parse(request.url).pathname;
    let content = route(pathname, handler, response);
    if (!content) {
      response.writeHead(404, {"Content-Type": "text/plain"});
      response.write("404 Not found");
      response.end();
    }
  }

  let port = process.env.port || 1337;
  http.createServer(onRequest).listen(port);
  console.log("Has been started.");
}
exports.start = start;
```

HTTP-methode beperken

```
if (request.method !== 'GET') {
  response.writeHead("405");
  response.end();
}
```

```
if (request.method !== 'POST') {
  response.writeHead("405");
  response.end();
}
```

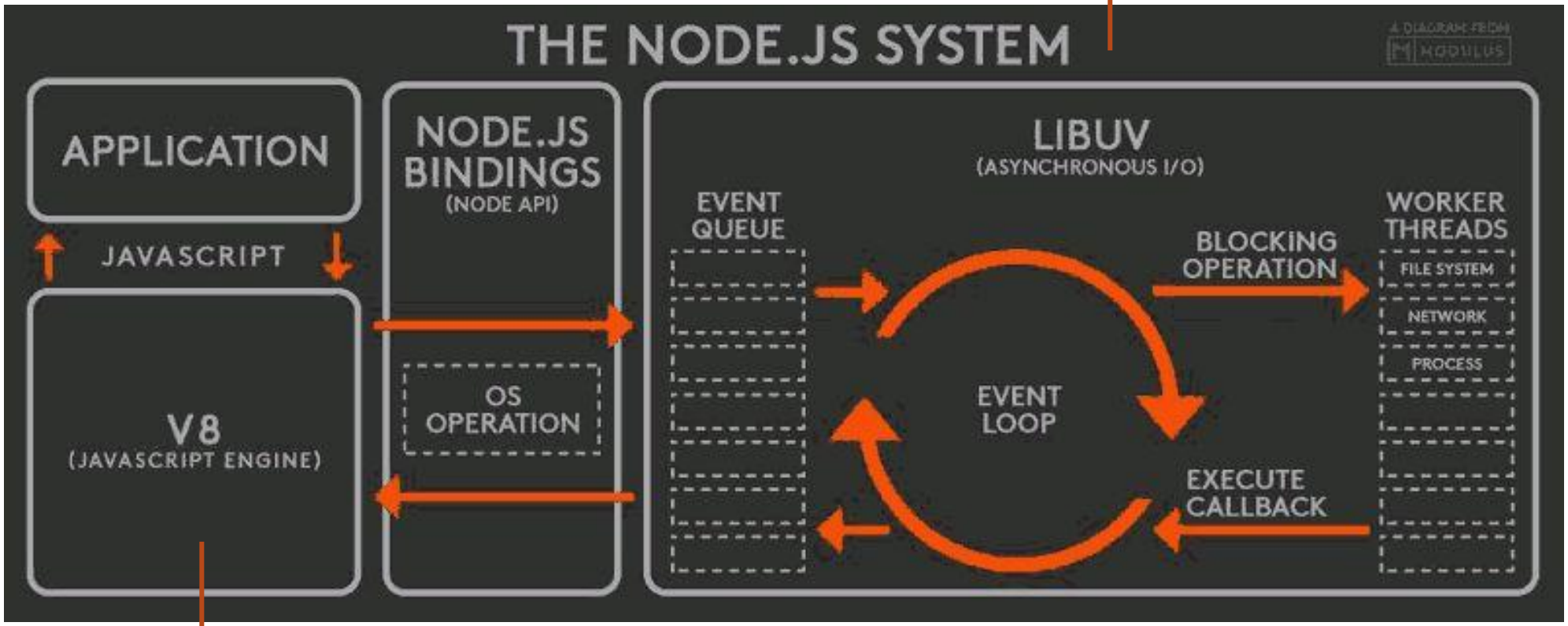
Overzicht

UNIVERSITEIT GENT

- Node.js en npm
- HTTP-server
- Routing
- Node.js versus Threads

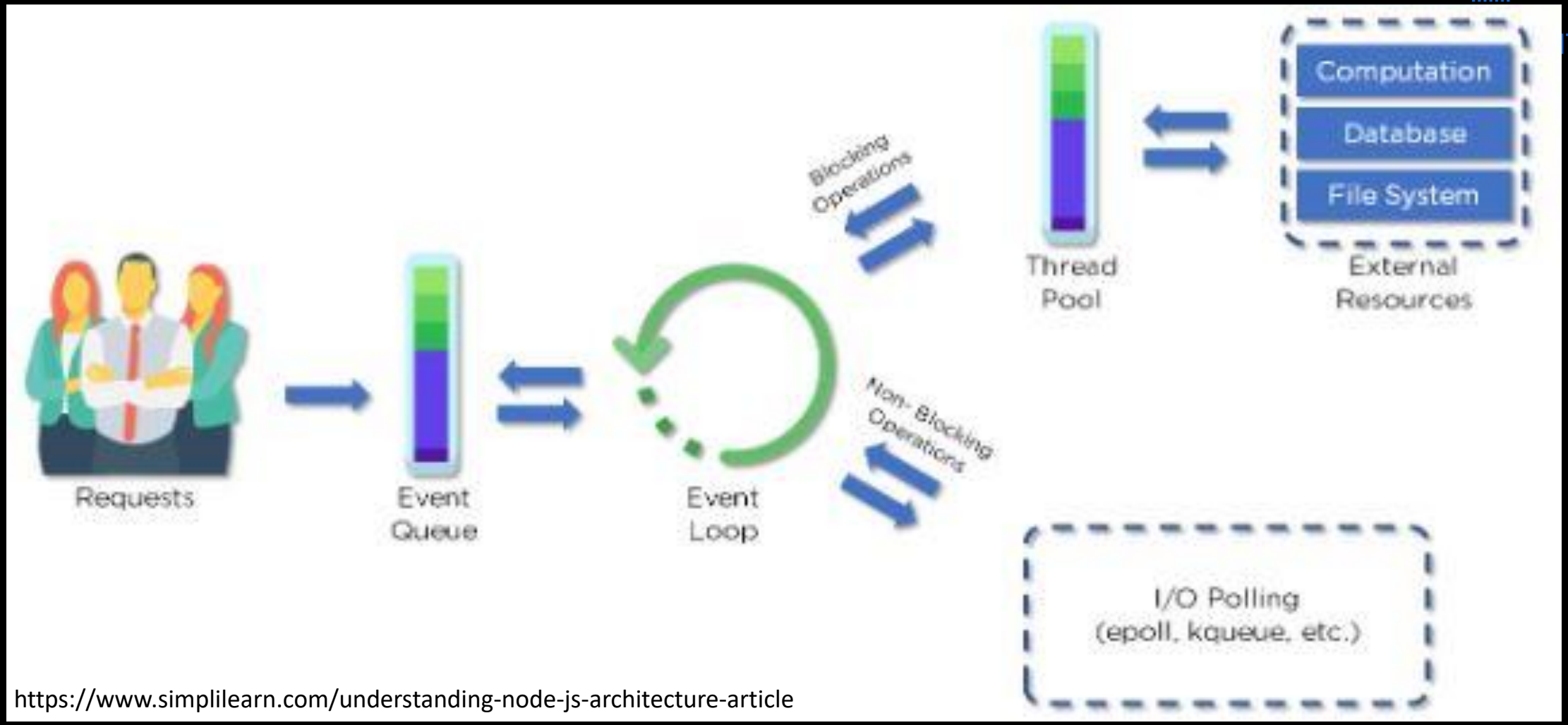
Node.js architectuur

eventbased: o.a. callbacks



https://www.vskills.in/certification/tutorial/node-js-architecture-2/

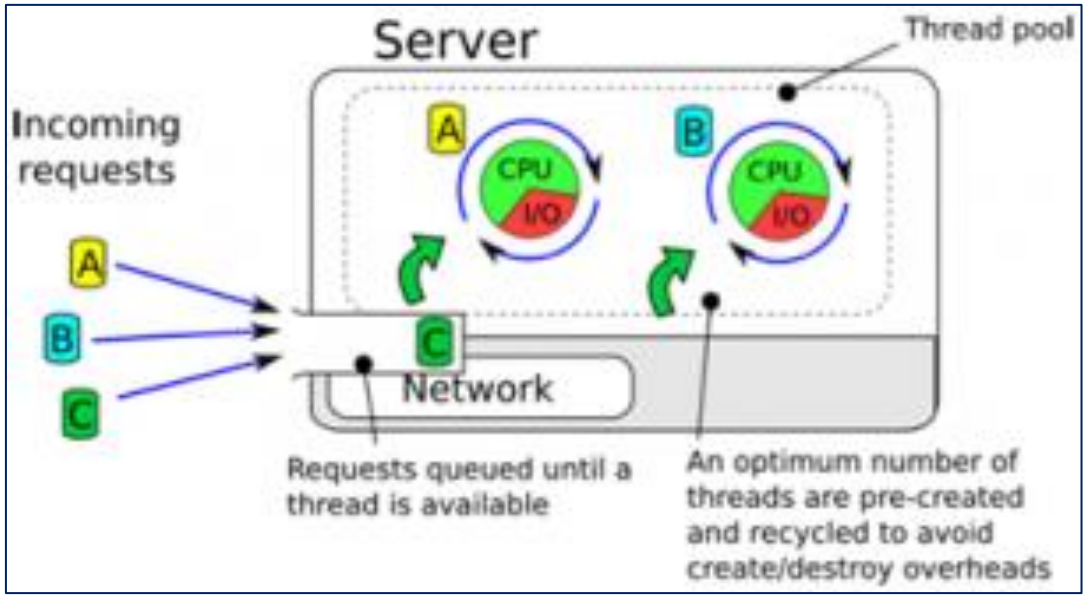
interpreteert JS-code

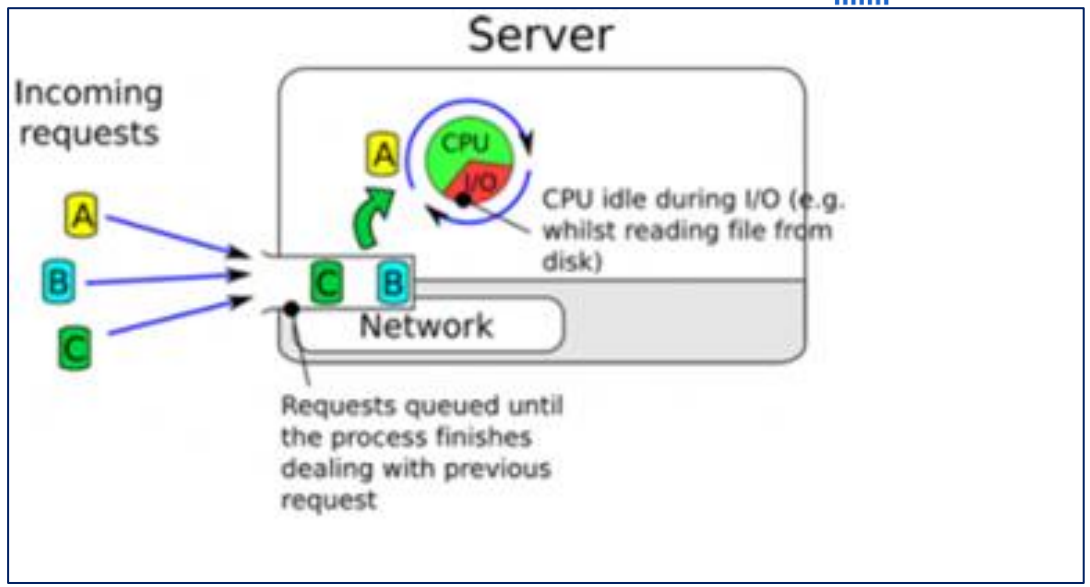


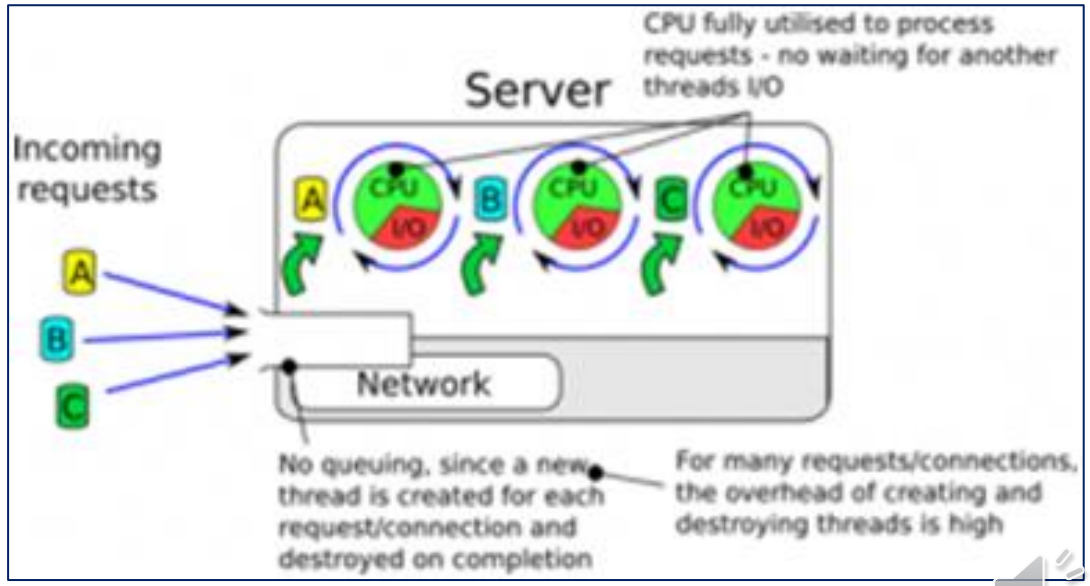
https://www.simplilearn.com/understanding-node-js-architecture-article

Node.js architectuur

Traditionele webserver - threads

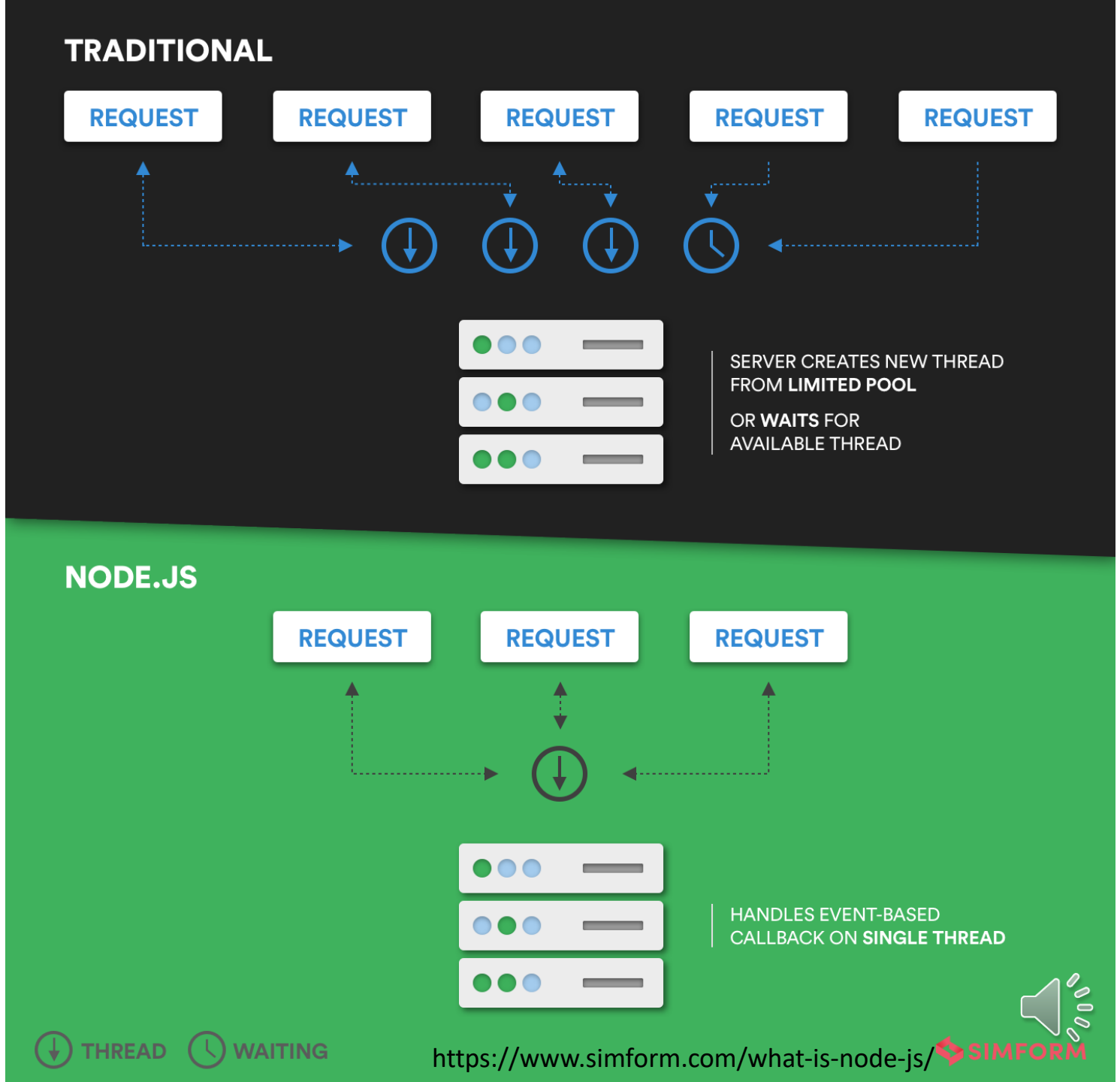






https://myshadesofgray.wordpress.com/2014/04/13/java-executor-framework/

Traditionele webserver versus Node.js



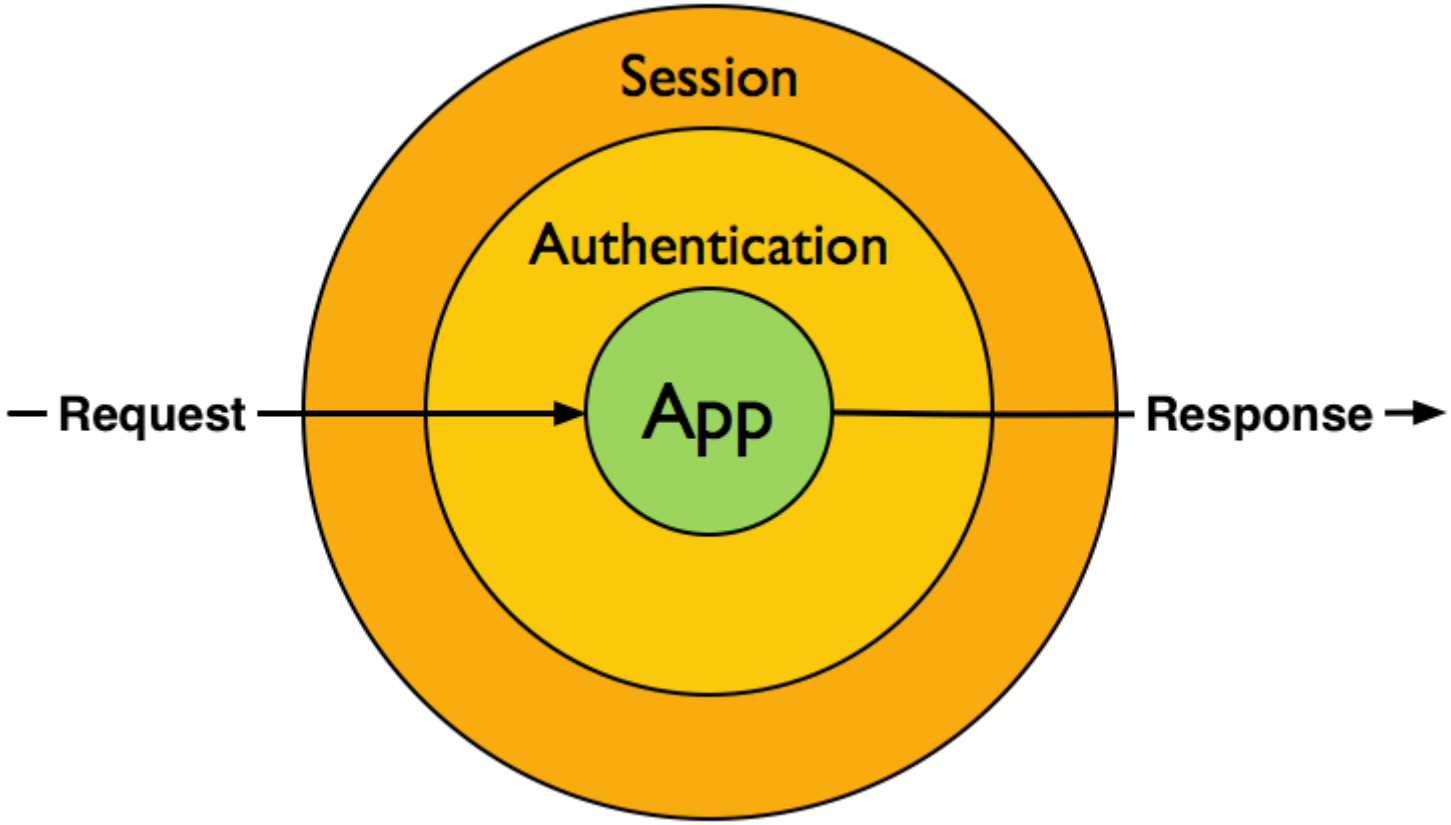
Aandachtspunten

- Applicatie verantwoordelijk voor “eerlijke” behandeling van de verschillende clients
  - Don't block the Event Loop
    - ➢ callbacks (then) korte uitvoeringstijd
    - ➢ Gebruik asynchrone methodes
    - ➢ Complexe berekeningen opdelen

Overzicht

- Node.js en npm
- HTTP-server
- Routing
- Node.js versus Threads
- Express
  - Inleiding

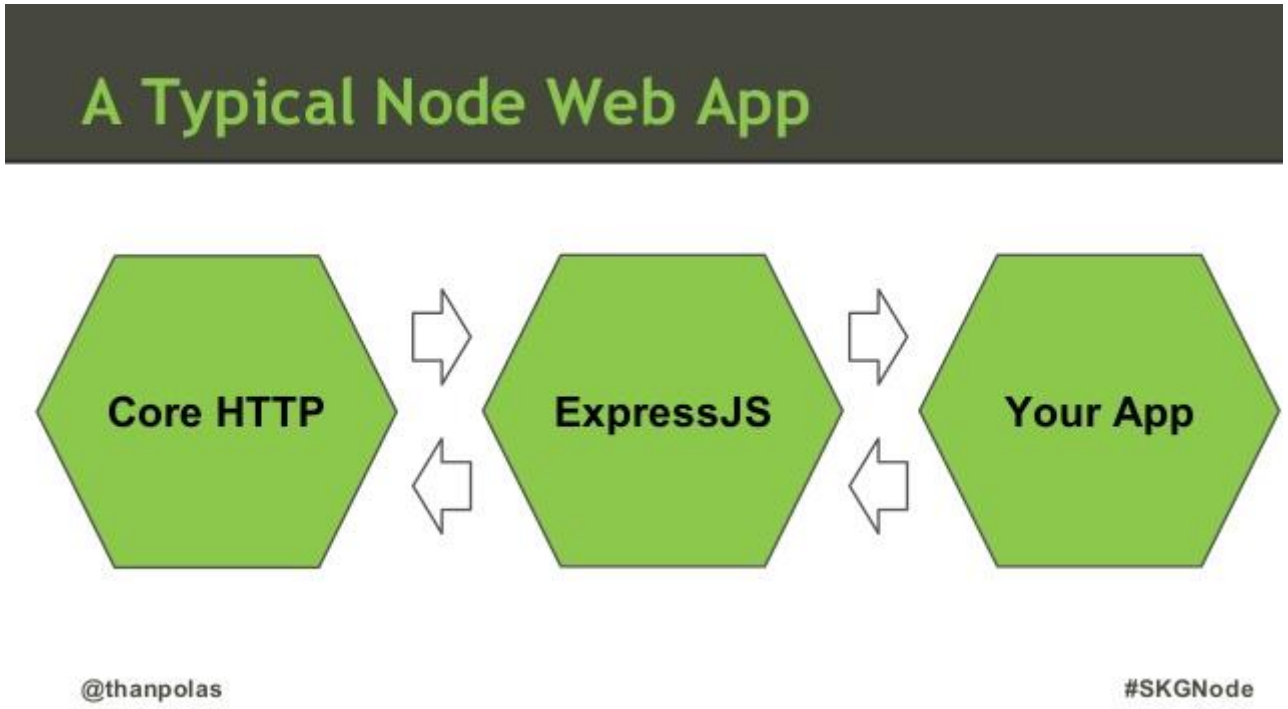
HTTP Middleware



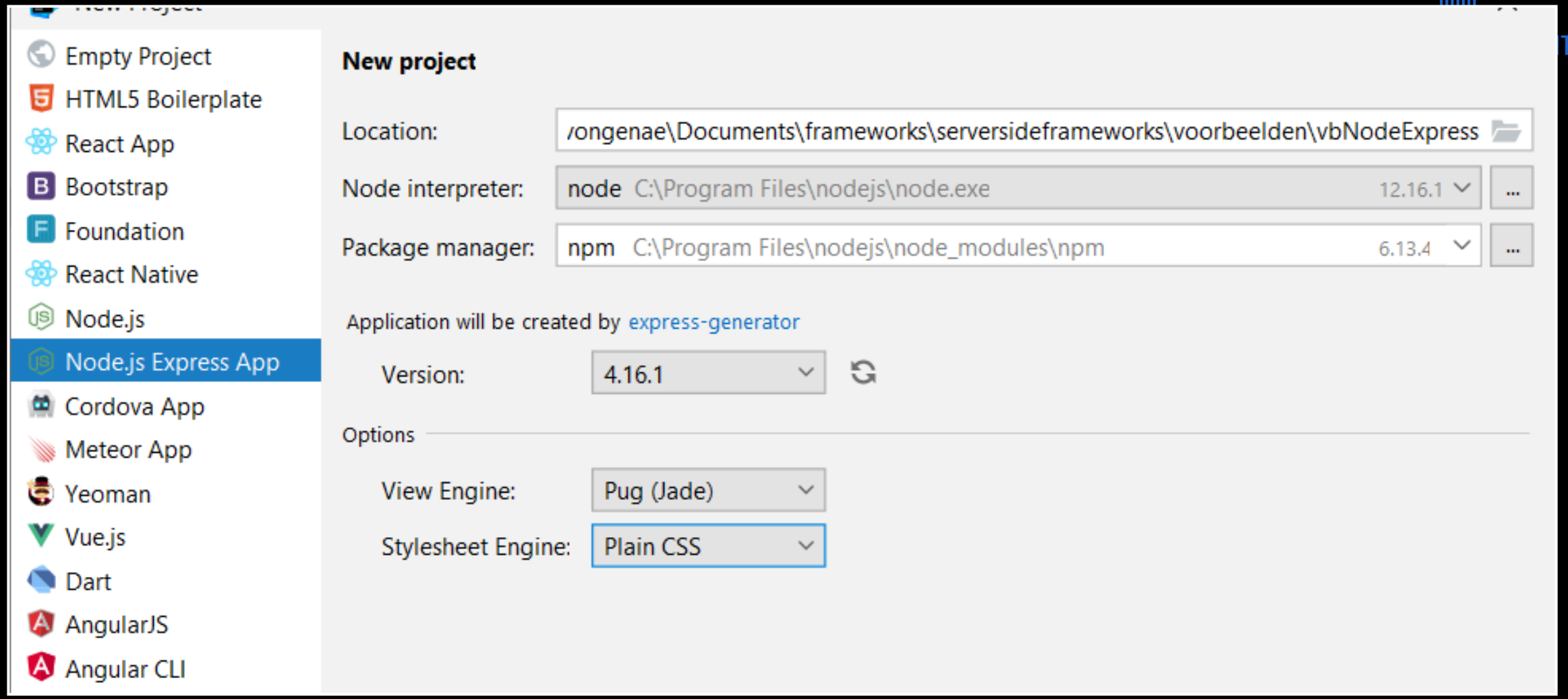
https://mattstauffer.com/blog/laravel-5.0-middleware-filter-style/

Express.js

- Framework om snel webapplicaties ontwikkelen in node.js



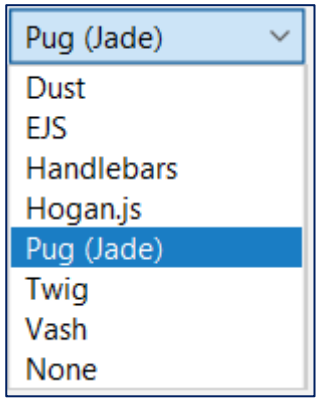
http://www.slideshare.net/thanpolas/intro-to-nodejs-39066435



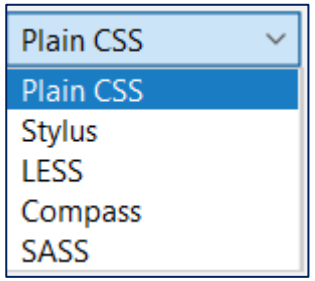
Applicatie maken (webstorm)

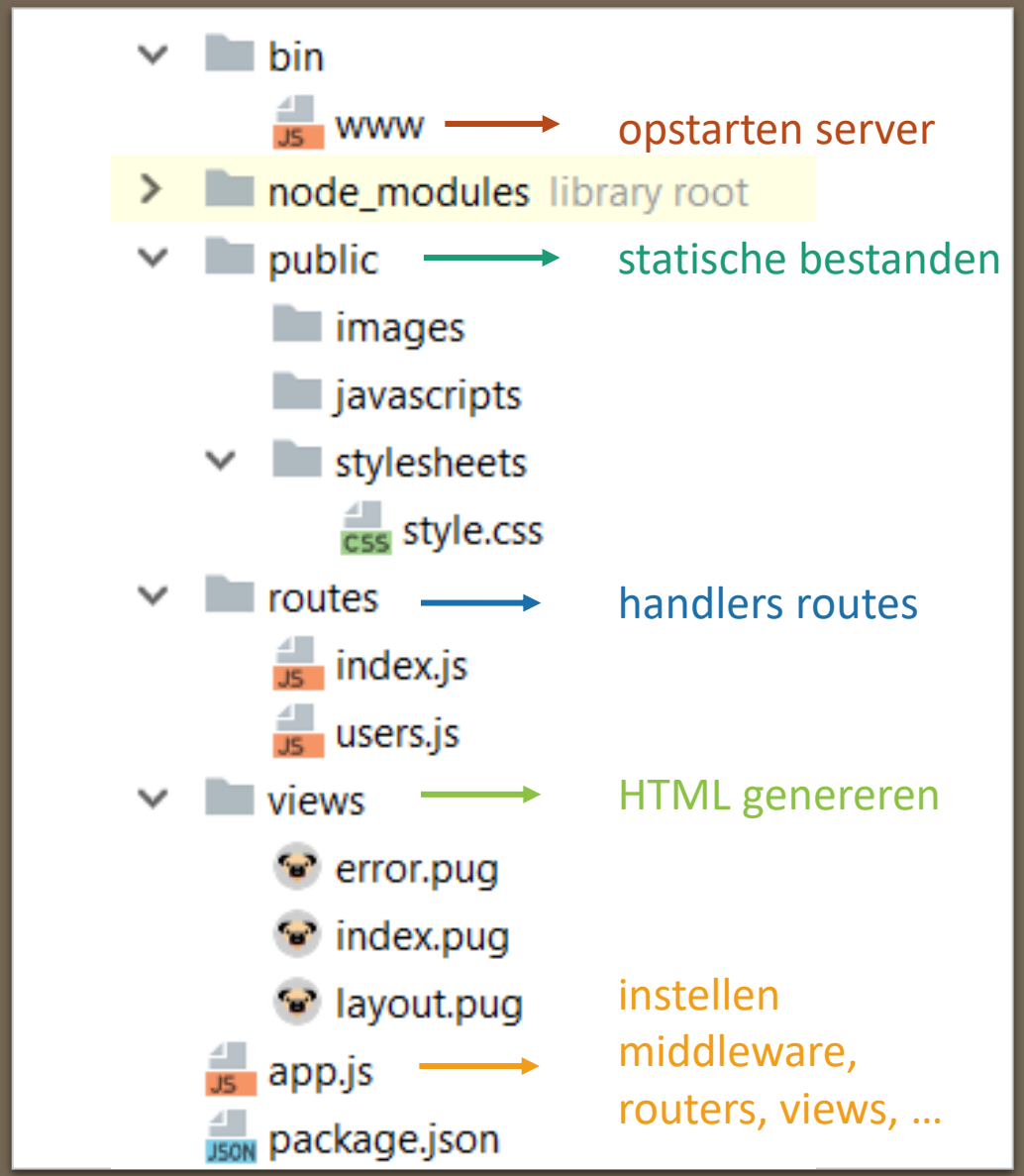
Installatie: opties

- View Engine
  - Pug



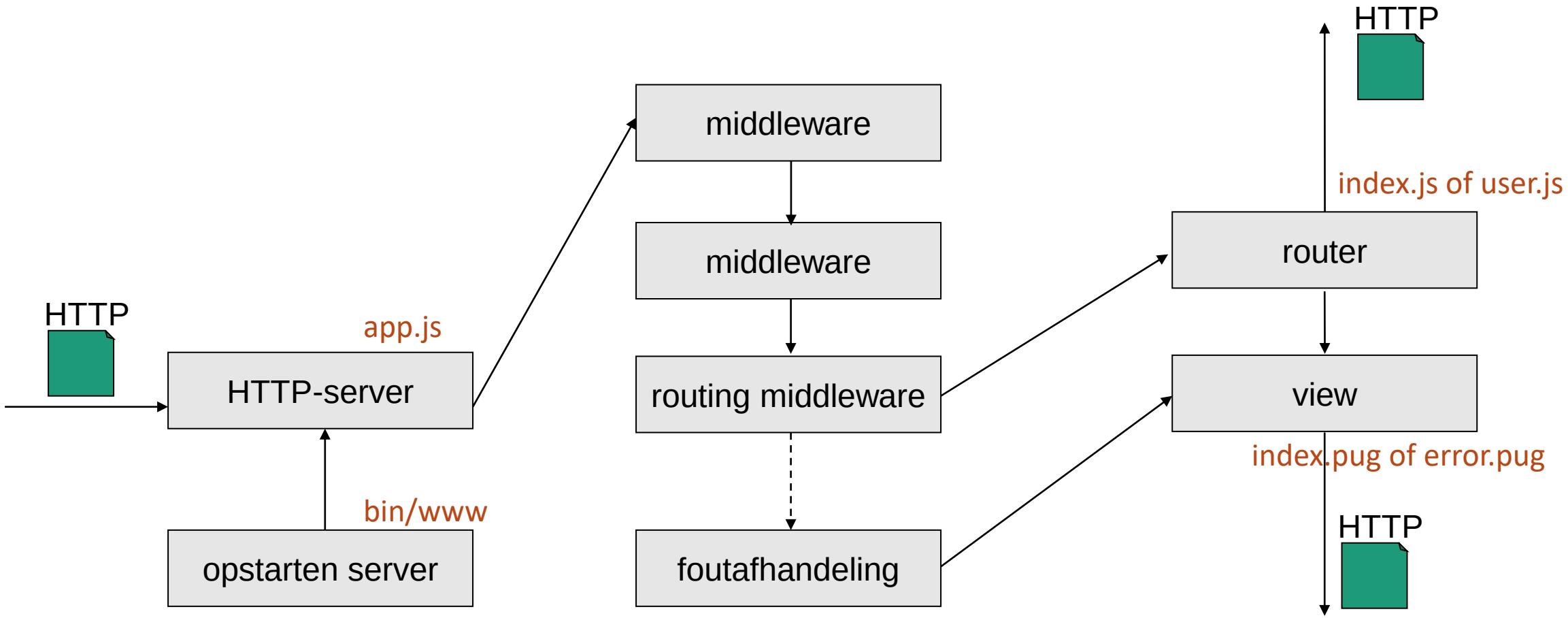
- Stylesheet
  - CSS





Structuur applicatie

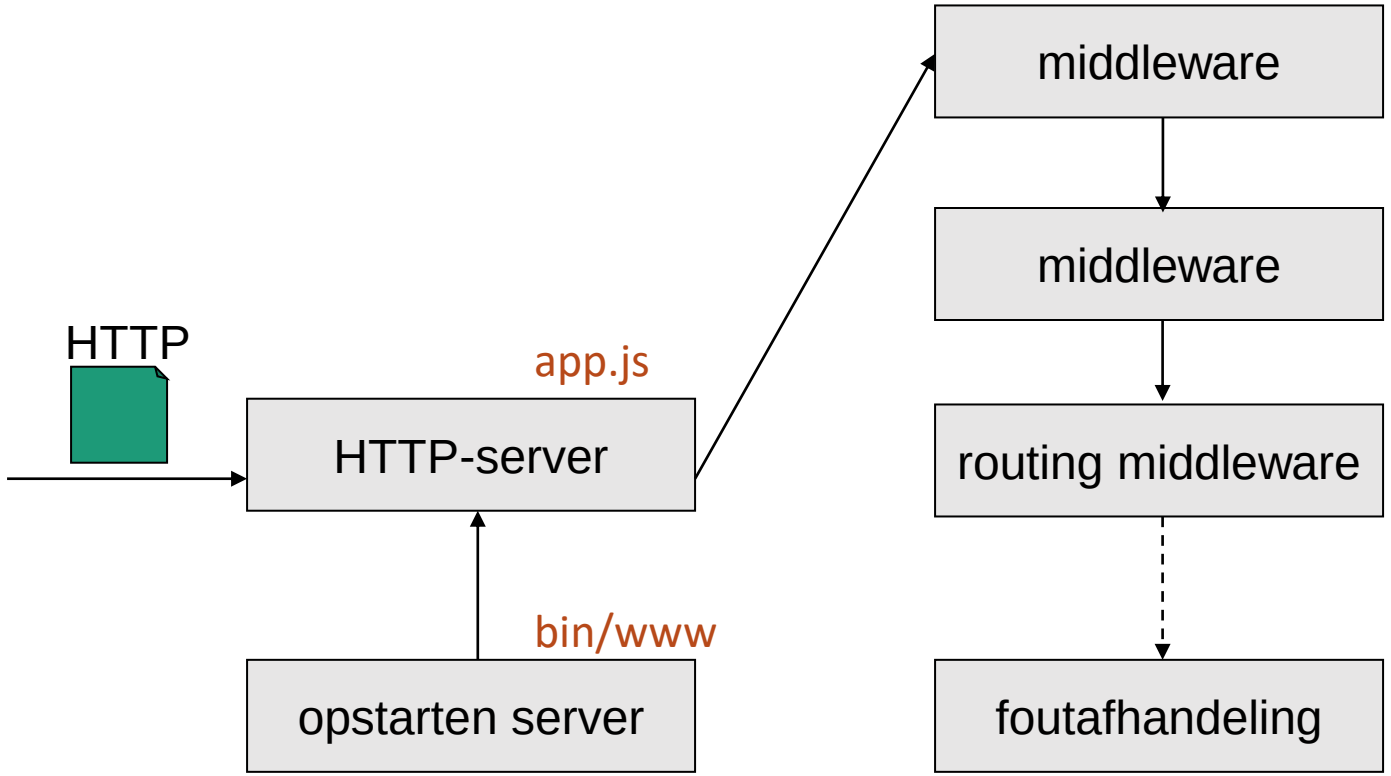
Structuur Express-applicaties



Overzicht

- Node.js en npm
- HTTP-server
- Routing
- Node.js versus Threads
- Express
  - Inleiding
  - app.js

Structuur Express-applicaties



app.js - modules

```
let express = require('express');
let path = require('path');
let favicon = require('serve-favicon');
let logger = require('morgan'); // logging
let cookieParser = require('cookie-parser');
let bodyParser = require('body-parser');

let routes = require('./routes/index');
let users = require('./routes/users');

let app = express();
```

gebruikte modules

zelfgeschreven routes

applicatie-object

app.js – middleware

```
// view engine setup
app.set('views', path.join(__dirname, 'views'));
app.set('view engine', 'pug');

// uncomment after placing your favicon in /public
//app.use(favicon(path.join(__dirname, 'public', 'favicon.ico')));
app.use(logger('dev'));
app.use(bodyParser.json());
app.use(bodyParser.urlencoded({ extended: false }));
app.use(cookieParser());
app.use(express.static(path.join(__dirname, 'public')));

app.use('/', routes);
app.use('/users', users);
```

views instellen

middleware

routes instellen

app.js – foutafhandeling

```
// catch 404 and forward to error handler
app.use(function(req, res, next) {
  let err = new Error('Not Found');
  err.status = 404;
  next(err);
});
```

enkel opgeroepen indien nog niet afgehandeld

doorsturen volgende middleware

app.js – foutafhandeling

```
// error handler
app.use(function(err, req, res, next) {
  // set locals, only providing error in development
  res.locals.message = err.message;
  res.locals.error =
    req.app.get('env') === 'development' ? err : {};

  // render the error page
  res.status(err.status || 500);
  res.render('error');
});

module.exports = app;
```

HTTP-statuscode instellen

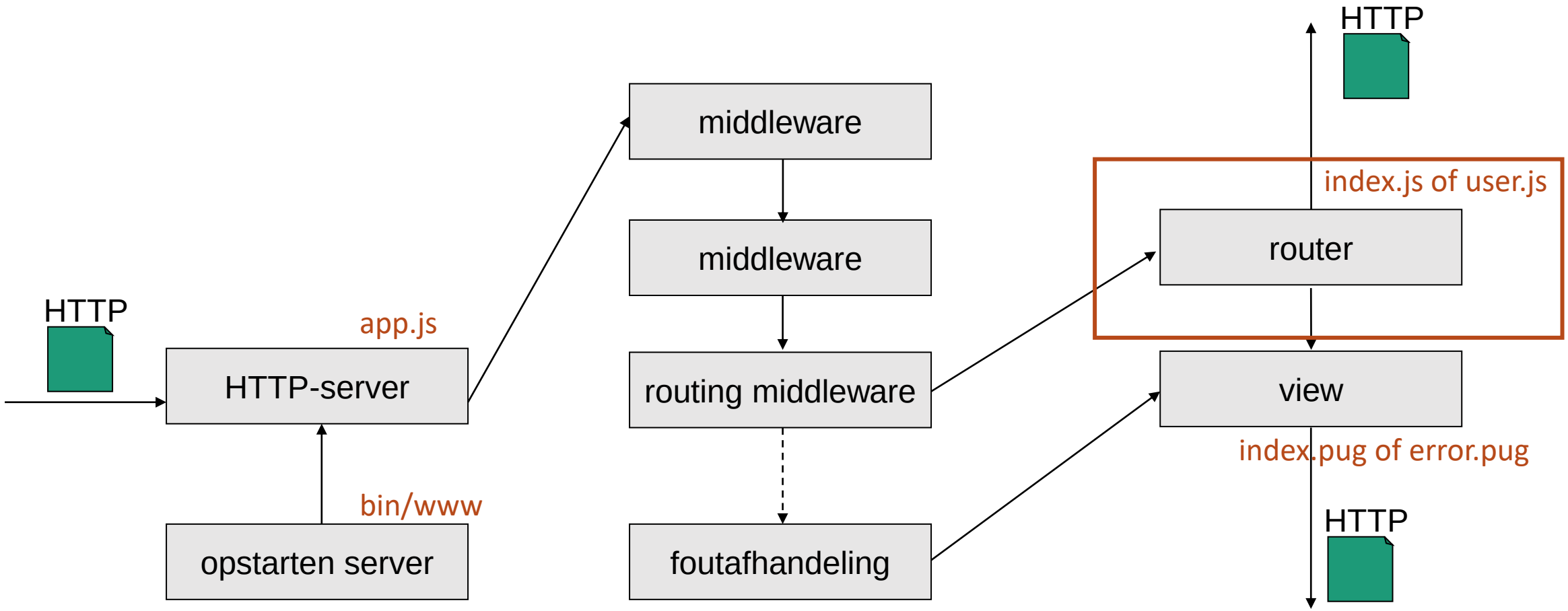
doorsturen naar view

applicatie-object exporteren

Overzicht

- Node.js en npm
- HTTP-server
- Routing
- Node.js versus Threads
- Express
  - Inleiding
  - app.js
  - Routing

Structuur Express-applicaties



routes/index.js

```
let express = require('express');
let router = express.Router();

/* GET home page. */
router.get('/', function(req, res, next) {
  res.render('index', { title: 'Express' });
});

module.exports = router;
```

router ophalen

route voor get naar / instellen (binnen huidig pad)

pad

functie die aanvraag afhandelt

doorsturen naar view

view

data voor view

routes/users.js

```
let express = require('express');
let router = express.Router();

/* GET users listing. */
router.get('/', function(req, res, next) {
  res.send('respond with a resource');
});

module.exports = router;
```

router ophalen

route voor get naar / instellen (binnen huidig pad)

pad

antwoord naar client

REST-service

Routing

UNIVERSITEIT GENT

- URL ↔ methode, module
- Single Page Applications (SPA)
  - URL ↔ REST call
- Non-SPA
  - URL ↔ volledige webpagina
- Module Express Router

```
let router = express.Router();
```


Pad ↔ router

```
app.use(SUB_PATH, ROUTER_MODULE)
```

app.js

```
let express = require('express’);
let adminRouter = require('./routes/adminRouter');
let app = express();
app.use('/admin', adminRouter);
```

adminRouter.js

```
let express = require('express');
let router = express.Router();

// The Admin-Site (http://localhost:3000/admin)
router.get('/', function(req, res) {
  res.send('Homepage of admin area!');
});
// The article-Site (http://localhost:3000/admin/article)
router.get('/article', function(req, res) {
  res.send('Show all articles!');
});
…
module.exports = router;
```

Express routing

- URI → actie
- Basis routing

```
app.METHOD(PATH, HANDLER)
```

```
let app = express();
app.get('/',  (req, res) => {…});
```

Verschillende handlers - parameters

```
router.get('/user/:id',function (req, res, next) {
  let id = req.params['id']
  console.log('CALLED ONLY ONCE');
  next();
});

router.get('/user/:id', function (req, res, next) {
  console.log('although this matches');
  next();
});

router.get('/user/:id', function (req, res) {
  console.log('and this matches too');
  res.end();
});
```

GET /user/42

```
CALLED ONLY ONCE
although this matches
and this matches too
```

route voor get

parameter in pad

request-object

response-object

volgende middleware

parameters

afsluiten antwoord

Functie route()

- Eén route (URL)
  - Verschillende handlers per HTTP-methode
    - GET
    - POST
    - PUT
    - DELETE

```
let router = express.Router();
router.route('/users/:user_id’)
  .all(function(req, res, next) {
    // runs for all HTTP verbs first
    req.user = {
      id: req.user_id,
      name: 'TJ’
    };
    next();
  })
  .get(function(req, res, next) {
    res.json(req.user);
  })
  .put(function(req, res, next) {
    req.user.name = req.params.name; // just an example of maybe updating the user
    // save user ... etc
    res.json(req.user);
  })
  .post(function(req, res, next) {
    next(new Error('not implemented'));
  })
  .delete(function(req, res, next) {
    next(new Error('not implemented'));
  });
```

route voor pad

voor alle HTTP-methodes

info toevoegen aan request

handler voor HTTP-methode

Routing - samenvatting

- Path ~ handler

```
router.METHOD(PATH, HANDLER)
```

- Deelpath ~ module

```
app.use(SUB_PATH, ROUTER_MODULE)
```

- Router-object

```
let router = express.Router();
```

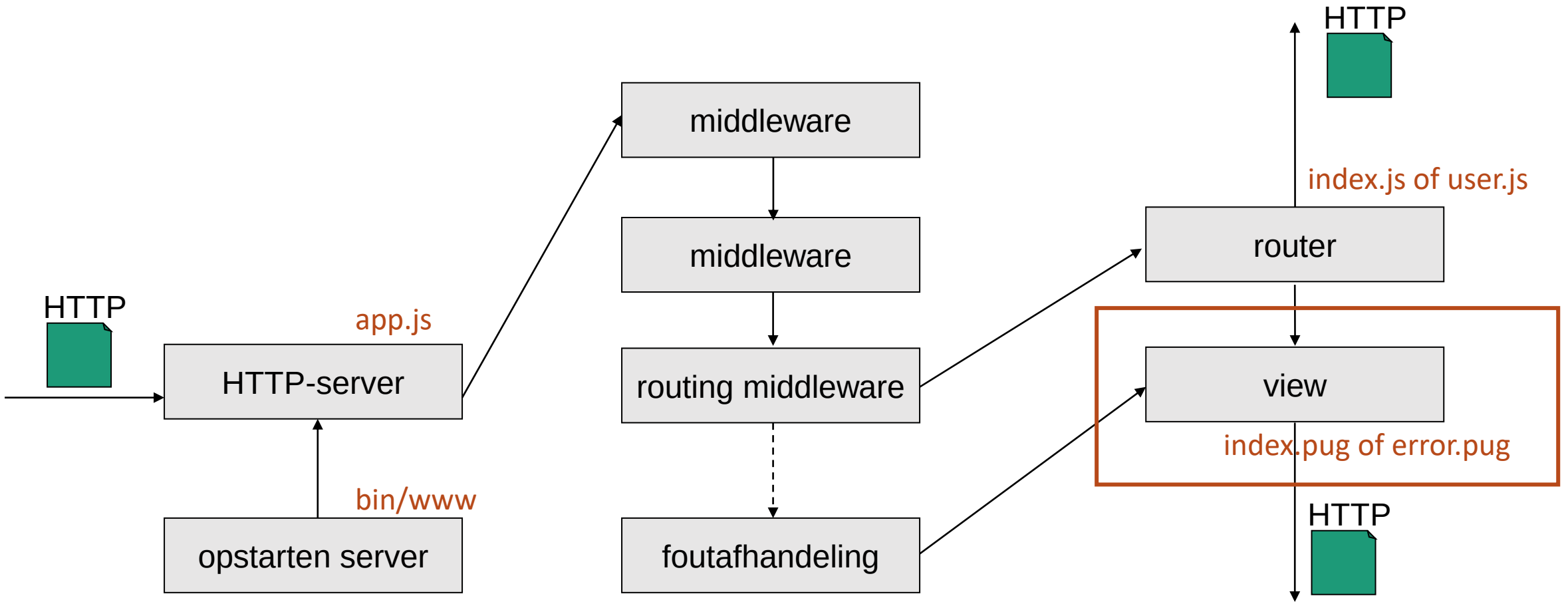
- Verschillende handlers voor één aanvraag: functie route
  - all(…), get(…), post(…), put(…), delete(…)
  - Doorsturen met next()

Overzicht

UNIVERSITEIT GENT

- Node.js en npm
- HTTP-server
- Routing
- Node.js versus Threads
- Express
  - Inleiding
  - app.js
  - Routing
  - Views

Structuur Express-applicaties



Views: layout.pug

- PUG
  - Template voor HTML
  - Tabs i.p.v. tags

```
doctype html
html
  head
    title= title
    link(rel='stylesheet', href='/stylesheets/style.css')
  body
    block content
```

in te vullen door specifiek pagina

Views: index.pug

```
extends layout

block content
  h1= title
  p Welcome to #{title}
```

maakt gebruik van layout

inhoud voor block uit layout

parameter meegegeven met render-methode

```
res.render('index', { title: 'Express' });
```

Gegenereerde HTML

```
<!DOCTYPE html>
<html>
  <head><title>Express</title>
     <link rel="stylesheet" href="/stylesheets/style.css">
  </head>
  <body>
    <h1>Express</h1>
    <p>Welcome to Express</p>
  </body>
</html>
```

Views: error.pug

```
extends layout

block content
  h1= message
  h2= error.status
  pre #{error.stack}
```

lokale omgevingsparameters

Overzicht

- Node.js en npm
- HTTP-server
- Routing
- Node.js versus Threads
- Express
  - Inleiding
  - app.js
  - Routing
  - Views