**# Cheat Sheet**

**!!! Да си направя скелет за проекта, в който после да добавя Public с css / images / views !!!**

# !!! Да видя в коя папка са static files в html-ите !!!

**1. Initialize project and structure**

**✔️ npm init –yes**

**✔️ Create > src/index.js**

**2. Setup dev environment**

**✔️ npm i -D nodemon**

**В package.json:**

"scripts": {

"start": "nodemon src/index.js"

},

**► ► ► TEST:   
В index.js:**

console.log('Server start')

**✔️ npm start**

**3. Install and setup express**

**✔️ npm i express**

**► ► ► TEST:**

**В index.js:**const express = require('express');

const app = express();

// Test action

app.get('/', (req, res) => {

res.send('First action');

});

app.listen(5000, console.log('Server is listening on port 5000...'));

// GO TO <http://localhost:5000/>

\* add static middleware

**✔️ Create > src/public**

**В index.js:**

// Middleware

app.use(express.static('public'));

**// \* add body parser**

**В index.js:**

// Body parser

app.use(express.urlencoded({extended: false}));

**// \* add routes**

**✔️ Create > src/routes.js**

**В routes.js:**

const router = require('express').Router();

// TODO: add controller router

module.exports = router;

**В index.js:**

const routes = require('./routes');

...

app.use(routes);

→ на мястото на // Test action.. app.get(‘/’,…..

→ В routes.js отива целия // Test action, като app.get става router.get

**► ► ► TEST:**

// GO TO <http://localhost:5000/>

**// 4. Add static resources**

**→ Вземаме от static папката и ги слагаме в нашата public**

**// 5. Add views folder with ready htmls**

**✔️ Create > src/views и в тях се добавят html-ите**

**// 6. Add express-handlebars view engine**

**// \* install**

**✔️ npm I express-handlebars**

**// \* add to express**

**В index.js:**

const handlebars = require('express-handlebars');

**// \* config extension**

**В index.js:**

// Add handlebars

app.engine('hbs', handlebars.engine({

extname: 'hbs'

}));

app.set('view engine', 'hbs');

**// \* config views folder (for src folder)**

app.set('views', 'src/views');

**// \* add main layout**

**✔️ Create > src/views/layouts – main.hbs**

**→ От home.html вземаме всичко с навигацията (obiknoweno do </header>) и го слагаме в main.hbs.**

**→ Вземаме и footer-a заедно с края.**

**--> Между тях слагаме**

{{{body}}}

**// \* add partials folder**

**✔️ Create > src/views/partials**

**// \* fix navigation to home**

**// \* render home page**

**В routes.js променяме на:**// Test action

router.get('/', (req, res) => {

res.render('home')

});

**И можем да променим home.html на home.hbs.**

**► ► ► TEST:**

// GO TO http://localhost:5000/

**// \* fix styles and images (static paths)**

**В index.js:**

const path = require('path');

…….

app.use('/static', express.static(path.resolve(\_\_dirname, 'public')));

**// 7. Add controllers folder with home controller**

**✔️ Create > src/controllers и в нея homeController.js.**

**→ Местя от routes.js тук:**const router = require('express').Router();

router.get('/', (req, res) => {

res.render('home');

});

module.exports = router;

--> В routes.js:

const router = require('express').Router();

const homeController = require('./controllers/homeController');

router.use(homeController);

module.exports = router;

**► ► ► TEST:**

// GO TO http://localhost:5000/

**// 8. Add database**

**// \* install mongoose**

**✔️ npm I mongoose**

**// \* connect database**

**В index.js:**

const mongoose = require('mongoose');

…

// TODO change DB name

mongoose.connect('mongodb://127.0.0.1:27017/petstagram')

.then(() => console.log('DB connected successfully'))

.catch(err => console.log('DB Error, ', err.message));

**// 9. Authentication**

**// \* add user controller**

**✔️ Create userController.js.**

**// \* add controller to routes.js**

*const* userController = require('./controllers/userController');

**...**

router.use('/users', userController);

**В userController.js:**

*const* router = require('express').Router();

router.get('/login', (req, res) *=>* {

res.render('users/login');

});

module.exports = router;

**// \* render login page**

**✔️ Create > src/views/users**

**--> login.html отива в нея и става login.hbs**

**--> Edit като остава само <main>...</main>**

**► ► ► TEST:**

// GO TO http://localhost:5000/users/login

**// \* fix header navigation to login and register**

Отиваме в main.hbs / header и оправяме линковете:

*<!--For Guest User-->*

<li *class=*"nav-item">

*<!-- Link to Login Page -->*

*<!-- TODO # -->*

<a *href=*"/users/login">

<i>Login</i>

</a>

</li>

<li *class=*"nav-item">

*<!-- Link to Register Page -->*

*<!-- TODO # -->*

<a *href=*"/users/register">

<i>Register</i>

</a>

...

*<!-- Link to Logout Page -->*

*<!-- TODO # -->*

<a *href=*"/users/logout">

<i>Logout</i>

</a>

...

<a *href=*"/">

<img *width=*"50px" *src=*"/static/images/free-30-instagram-stories-icons23\_122570.png" *alt=*"img1">

</a>

...

<a *class=*"home" *href=*"/">

<i>Petstagram</i>

</a>

...

**//    \* render register page**

**В userController.js:**

**router.get('/register', (req, res) *=>* {**

res.render('users/register');

});

**✔️ View-то:**

**--> register.html отива в нея и става register.hbs**

**--> Edit като остава само <main>...</main>**

**► ► ► TEST:**

**// GO TO** [**http://localhost:5000/users/r**](http://localhost:5000/users/login)**egister**