# Node.js Web Server Exercises

## Movie DB server

### 1.1 Create Node server

Create Node web server and add **index**, **favicon** and static files handlers, you may use the provided example from the last lecture or try to build it from the ground.

### 1.2 Restrict the static file handler

Restrict the handler to serve only files from the '**content**' folder. Restrict it to serve only **html**, **css, js, png and jpg** files.

### 1.3 Create file handler that displays all movies from the DB

The server should return HTML containing all images, sorted by their release year, with links to their details page.

**HINT**: The **HTML** should be generated **runtime** string depending on the current saved images, you may also use the provided 'viewAll.html'.

For displaying the movies data, you may replace the following code in the provided 'viewAll.html':

<div id="replaceMe">{{replaceMe}}</div>

For each movie: <div class="movie">

<img class="moviePoster" src="{{Your movie poster URL}}"/>

</div>

**Video: 01:51:30** - Decode data from Database(**decodeUriComponenen(m.moviePoster**))

### 1.4 Create file handler for adding a movie to the DB

Users should be able to fill title, URL of an image, year of release and synopsis of a movie, then send the data to the server. The request should be **POST**. The server should save the data in the provided **db object**. If the name or the URL are null or empty, the server should return a friendly error message.

For displaying the error message, you may replace the following code in the provided 'addMovie.html':

<div id="replaceMe">{{replaceMe}}</div>

on Error: <div id="errBox"><h2 id="errMsg">Please fill all fields</h2></div>

on Success: <div id="succssesBox"><h2 id="succssesMsg">Movie Added</h2></div>

**HINT**: Search online how to parse the form data on the server.

### 1.5 Create details page

Each movie should have a details page where the movie poster and data is shown.

**HINT**: The URL may be something like **'/movies/details/4'** where **4** should be extracted and it shows the index of the image to show, you may use the provided 'details.html'.

For displaying single details about a movie, you may replace the following code in the provided 'details.html':

<div id="replaceMe">{{replaceMe}}</div>

Replacement: <div class="content">

<img src="{{Your movie poster URL}}" alt=""/>

<h3>Title {{Your movie title}}</h3>

<h3>Year ${Your movie release year}</h3>

<p> {{Youer movie description}}</p>

</div>​

### 1.6 Add menus

Add **links** to all HTML pages.

### 1.7 Write a header handler

Write a handler that catches a header with name "**StatusHeader**" and with value "**Full**" and, if such is present to return a "**status.html**" file. That HTML file should show the total number of movies.

## Use GULP to minify the HTML

Minify the served HTML with a GULP task.