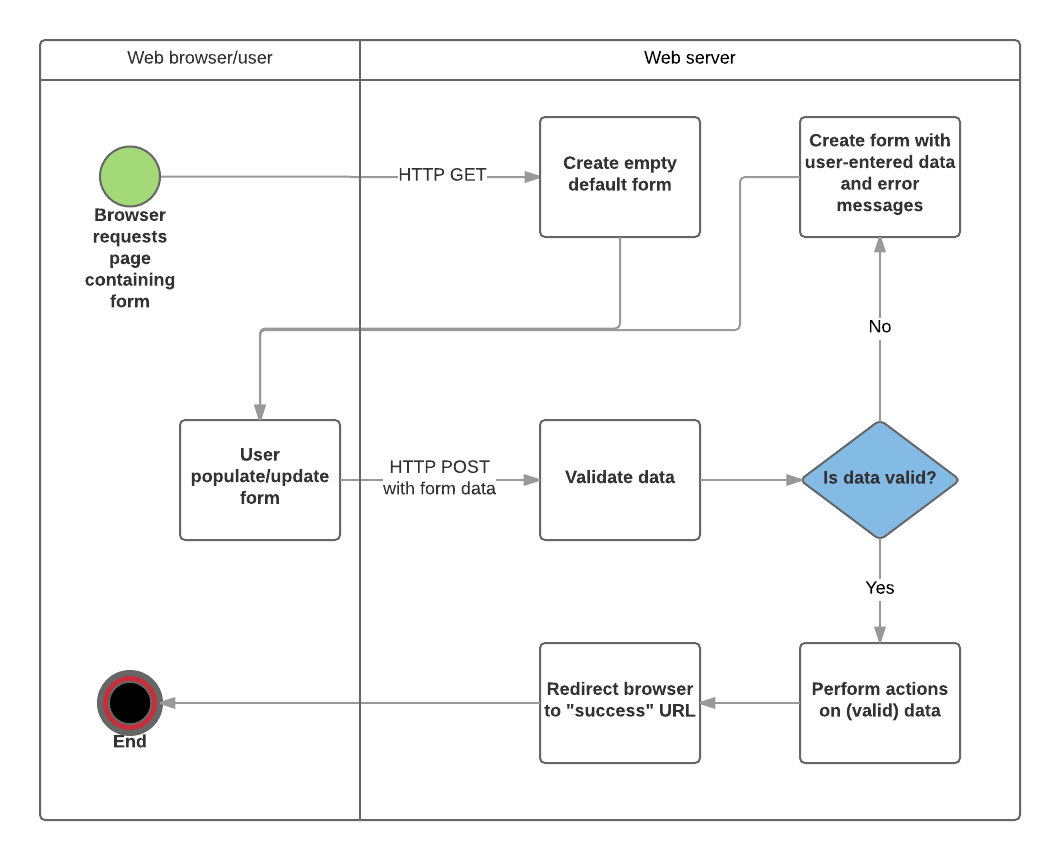
Робота з формами

Передача даних через get запит.



|  |  |  |
| --- | --- | --- |
| Дані, які передані через get запит доступні у об’єкті request через властивості query | Форма | Обробка даних з форми |
| у запиті  / шлях ? **власт.1**=зн.1 & **власт.2**=зн.2 ...  при обробці  req.**query**. властивість1   * The GET method should only be used for forms that don't change user data (e.g. a search form). It is recommended for when you want to be able to bookmark or share the URL. | <!DOCTYPE **html**> <**html lang="en"**> <**head**>  <**meta charset="UTF-8"**>  <**title**>Title</**title**> </**head**> <**body**> <**form action="/data" method="get"**>  <**label**>  User name  <**input type="text" name="uName"**>  </**label**>  <**label**>  User age  <**input type="number" name="uAge"**>  </**label**>  <**input type="submit" value="Надіслати"**> </**form**> </**body**> </**html**> | **--- Дані передані через запит ---**  /data? **uName**=Ivan & **uAge**=34  **---------------------------------**  **const** express=require(**"express"**); **var** app=express(); app.use(express.**static**(\_\_dirname+**'/public'**));  app.get(**"/data"**,**function** (req,res) {  **console**.log(req.**query.uName**);  **console**.log(req.**query**.**uAge**);  });   app.listen(3000,**function** () {  **console**.log(**"Listen 3000"**); }) |
| Передача масивів | <!DOCTYPE **html**> <**html lang="en"**> <**head**>  <**meta charset="UTF-8"**>  <**title**>Title</**title**> </**head**> <**body**> <**form action="/dataArr" method="get"**>  <**label**>  User1 name<**input type="text" name="uName"**>  </**label**>  <**label**>  User2 name<**input type="text" name="uName"**>  </**label**>  <**label**>  User3 name<**input type="text" name="uName"**>  </**label**>  <**label**>  User age<**input type="number" name="uAge"**>  </**label**>  <**input type="submit" value="Надіслати"**> </**form**> </**body**> </**html**> | **const** express=require(**"express"**); **var** app=express(); **const** bodyParser = require(**"body-parser"**); **const** urlencodedParser = bodyParser.urlencoded({**extended**: **false**});  app.use(express.**static**(\_\_dirname+**'/public'**));    app.get(**"/dataArr"**,**function** (req,res) {  **console**.log(req.**url**);  **console**.log(req.**query**.**uName**);  **console**.log(req.**query**.**uAge**); });   app.listen(3000,**function** () {  **console**.log(**"Listen 3000"**); })  //----------------------    /dataArr?uName=a11&uName=a22&uName=a333&uAge=33  [ 'a11', 'a22', 'a333' ]  33 |
| Передача об’єктів | <!DOCTYPE **html**> <**html lang="en"**> <**head**>  <**meta charset="UTF-8"**>  <**title**>Title</**title**> </**head**> <**body**> <**form action="/data" method="get"**>  <**label**>  User name  <**input type="text" name="uName"**>  </**label**>  <**label**>  User age<  **input type="number" name="uAge"**>  </**label**>  <**input type="submit" value="Надіслати"**> </**form**> <**script**>  //=========================================== **document**.querySelector(**"[type='submit']"**).**onclick**=**function** (e) {  e.preventDefault(); **var** user={  **uName**:**document**.getElementsByName(**"uName"**)[0].**value**,  **uAge**:**document**.getElementsByName(**"uAge"**)[0].**value** };  **var** request = **new** XMLHttpRequest();  request.open(**"POST"**, **"/dataObj"**, **true**);  request.setRequestHeader(**"Content-Type"**, **"application/json"**);  request.addEventListener(**"load"**, **function** () {   alert(**"Ok"**); });  request.send(***JSON***.stringify(user)); } </**script**> </**body**> </**html**> | **const** express=require(**"express"**); **var** app=express();   **const** jsonParser = express.**json**();   app.post(**"/dataObj"**,  jsonParser,  **function** (req,res) {  **console**.log(**"Objj"**);  **console**.log(req.**body**);  **console**.log(req.**body**.**uName**);  **console**.log(req.**body**.**uAge**);  res.send(**"ok"**); });   app.listen(3000,**function** () {  **console**.log(**"Listen 3000"**); }) |

Передача даних через post

|  |  |  |
| --- | --- | --- |
| Дані, які передані через post запит  npm install body-parser --save | Форма | Обробка даних з форми |
| **const** bodyParser = require(**"body-parser"**); **const** urlencodedParser =  bodyParser.urlencoded({**extended**: **false**});  app.post(**"/шлях"**, urlencodedParser,  **function** (req,res) {  req.**body**. властивість ;  });   * The POST method should always be used if the data is going to result in a change to the server's database, because this can be made more resistant to cross-site forgery request attacks. | <!DOCTYPE **html**> <**html lang="en"**> <**head**>  <**meta charset="UTF-8"**>  <**title**>Title</**title**> </**head**> <**body**> <**form action="/data" method="post"**>  <**label**>  User name  <**input type="text" name="uName"**>  </**label**>  <**label**>  User age  <**input type="number" name="uAge"**>  </**label**>  <**input type="submit" value="Надіслати"**> </**form**> </**body**> </**html**> | **const** express=require(**"express"**); **var** app=express(); **const** bodyParser = require(**"body-parser"**); **const** urlencodedParser = bodyParser.urlencoded({**extended**: **false**});  app.use(express.**static**(\_\_dirname+**'/public'**));  app.post(**"/data"**,  urlencodedParser,  **function** (req,res) {  **console**.log(req.**url**);  **console**.log(req.**body**.**uName**);  **console**.log(req.**body**.**uAge**); });  app.listen(3000,**function** () {  **console**.log(**"Listen 3000"**); }) |

==========================================================================================================

==========================================================================================================

==========================================================================================================

Обробка даних з використанням formidable (<https://github.com/felixge/node-formidable>)

npm i -S formidable

|  |  |  |
| --- | --- | --- |
| Аналізуємо передані значення полів форми. Вказуємо шлях до директорії, де зберегти файли (імена формуються автоматично) | | |
| Форма | Загальна схема | **Приклад** |
| <!DOCTYPE html> <html lang="en"> <head>  <meta charset="UTF-8">  <title>Title</title> </head> <body> <form action="/data" enctype="multipart/form-data" method="post">  <label>  User name  **<input type="text" name="uName">**  </label>  <label>  User age  **<input type="number" name="uAge">**  </label>  <label>  User foto  **<input type="file" name="uFoto" multiple>**  </label>  <input type="submit" value="Надіслати"> </form> </body> </html> | **formidable** = **require**(**'formidable'**);   app.post(**'/шлях'**, **function** (req, res){  //Створюємо об’єкт IncomingForm **var** form = **new formidable**.**IncomingForm**();  //Вказуємо куди завантажувати файли form.**uploadDir**=\_\_dirname + **'/uploads/'**;  . . . . . . . . .  form.parse(req, **function**(err, fields, files) {  . . . . . .  //fields – об’єкт, що містить поля  //files – об’єкт, що міст. інф. про файл  . . . . . .  }); | **formidable** = **require**(**'formidable'**); **var** util=**require**(**"util"**);  app.post(**'/data'**, **function** (req, res){  **var** form = **new formidable**.**IncomingForm**();  form.**uploadDir**=\_\_dirname + **'/uploads/'**;   form.parse(req, **function**(err, fields, files) {  res.writeHead(200, {**'content-type'**: **'text/plain'**});  res.write(**'received upload:\n\n'**);  res.end(util.inspect({**fields**: fields, **files**: files}));  }); });  //--------------------------- |

|  |  |  |
| --- | --- | --- |
| Аналізуємо події (https://github.com/felixge/node-formidable) | | |
|  |  |  |
|  | **formidable** = **require**(**'formidable'**);  app.post(**'/data'**, **function** (req, res){ **var** form = **new formidable**.IncomingForm();  form.parse(req);  // Для події почату завантаження файлу  form.on(**'fileBegin'**, **function** (name, file){  . . .  }  //Файл завантажено  form.on(**'file'**, **function** (name, file){  . . .  }  . . . .  });  . . . . . | **formidable** = **require**(**'formidable'**); **var** util=**require**(**"util"**);  app.post(**'/data'**, **function** (req, res){ **var** form = **new formidable**.IncomingForm();  form.parse(req);  **var** filesCount=0;  // Для події почату завантаження файлу (задаємо де збергти)  form.on(**'fileBegin'**, **function** (name, file){  **console**.log(file);  file.**path** = \_\_dirname + **'/uploads/'** + file.**name**;  filesCount++;  });  //Файл завантажено  form.on(**'file'**, **function** (name, file){  **console**.log(**'Uploaded '** + file.**name**);  });  //Закінчено завантаження усіх файлів  form.on(**'end'**, **function**() {  res.end(**'Uploaded file'** + filesCount);  }); }); |

Приклад. Здійснюємо обробку і полів, і файлів

|  |  |  |
| --- | --- | --- |
| Аналізуємо і поля і події (https://github.com/felixge/node-formidable) | | |
|  |  |  |
|  | **formidable** = **require**(**'formidable'**);  app.post(**'/data'**, **function** (req, res){ **var** form = **new formidable**.IncomingForm();  form.parse(req, **function** (err, fields, files) {  **//Аналіз полів**  . . . .  }).on(**'fileBegin'**, **function** (name, file){  . . .  }.on(**'file'**, **function** (name, file){  . . .  }  . . . . . | **formidable** = **require**(**'formidable'**)  app.post(**'/data'**, **function** (req, res){  **var** form = **new formidable**.IncomingForm();  **var** filesCount=0; form.parse(req, **function** (err, fields, files) {  **//Аналіз полів**  **console**.log(fields.**uName**);  **console**.log(fields.**uAge**); }).on(**'fileBegin'**, **function** (name, file){ **//Аналіз подій**  **console**.log(file);  file.**path** = \_\_dirname + **'/uploads/'** + file.**name**;  filesCount++;  });  form.on(**'file'**, **function** (name, file){  **console**.log(**'Uploaded '** + file.**name**);  });  form.on(**'end'**, **function**() {  res.end(**'Uploaded file'** + filesCount);  });  }); |

Валідація форм

<https://developer.mozilla.org/en-US/docs/Learn/Server-side/Express_Nodejs/forms>

### Validation and sanitization[Section](https://developer.mozilla.org/en-US/docs/Learn/Server-side/Express_Nodejs/forms#Validation_and_sanitization)

Before the data from a form is stored it must be validated and sanitized:

* Validation checks that entered values are appropriate for each field (are in the right range, format, etc.) and that values have been supplied for all required fields.
* Sanitization removes/replaces characters in the data that might potentially be used to send malicious content to the server.

For this tutorial, we'll be using the popular [express-validator](https://www.npmjs.com/package/express-validator) module to perform both validation and sanitization of our form data.

#### Installation

Install the module by running the following command in the root of the project.

npm install express-validator

Приклад

|  |  |  |
| --- | --- | --- |
|  | Форма | Опрацювання форми |
|  | extends layout  block content  h1=title  form(method='POST' action='')  div.form-group  label(for='first\_name') First Name:  input#first\_name.form-control(type='text' placeholder='First name (Christian) last' name='first\_name' required='true' value=(undefined===author ? '' : author.first\_name) )  label(for='family\_name') Family Name:  input#family\_name.form-control(type='text' placeholder='Family name (surname)' name='family\_name' required='true' value=(undefined===author ? '' : author.family\_name))  div.form-group  label(for='date\_of\_birth') Date of birth:  input#date\_of\_birth.form-control(type='date' name='date\_of\_birth' value=(undefined===author ? '' : author.date\_of\_birth) )  button.btn.btn-primary(type='submit') Submit  **if errors**  **ul**  **for error in errors**  **li!= error.msg** | // Display Author create form on GET.  exports.author\_create\_get = function(req, res, next) {  res.render('author\_form', { title: 'Create Author'});  }; Controller—post route[Section](https://developer.mozilla.org/en-US/docs/Learn/Server-side/Express_Nodejs/forms/Create_author_form#Controller%E2%80%94post_route) Find the exported author\_create\_post() controller method, and replace it with the following code.  const { body,validationResult } = require('express-validator/check');  const { sanitizeBody } = require('express-validator/filter');  // Handle Author create on POST.  exports.author\_create\_post = [  // Validate fields.  body('first\_name').isLength({ min: 1 }).trim().withMessage('First name must be specified.')  .isAlphanumeric().withMessage('First name has non-alphanumeric characters.'),  body('family\_name').isLength({ min: 1 }).trim().withMessage('Family name must be specified.')  .isAlphanumeric().withMessage('Family name has non-alphanumeric characters.'),  body('date\_of\_birth', 'Invalid date of birth').optional({ checkFalsy: true }).isISO8601(),  body('date\_of\_death', 'Invalid date of death').optional({ checkFalsy: true }).isISO8601(),  // Sanitize fields.  sanitizeBody('first\_name').trim().escape(),  sanitizeBody('family\_name').trim().escape(),  sanitizeBody('date\_of\_birth').toDate(),  sanitizeBody('date\_of\_death').toDate(),  // Process request after validation and sanitization.  (req, res, next) => {  // Extract the validation errors from a request.  const errors = validationResult(req);  if (!errors.isEmpty()) {  // There are errors. Render form again with sanitized values/errors messages.  res.render('author\_form', { title: 'Create Author', author: req.body, errors: errors.array() });  return;  }  else {  // Data from form is valid.  // Create an Author object with escaped and trimmed data.  var author = new Author(  {  first\_name: req.body.first\_name,  family\_name: req.body.family\_name,  date\_of\_birth: req.body.date\_of\_birth,  date\_of\_death: req.body.date\_of\_death  });  author.save(function (err) {  if (err) { return next(err); }  // Successful - redirect to new author record.  res.redirect(author.url);  });  }  }  ]; |