

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ

ІМЕНІ ІГОРЯ СІКОРСЬКОГО”

Факультет прикладної математики

Кафедра програмного забезпечення комп’ютерних систем

**Лабораторна робота №** 3

з дисципліни “Основи web-програмування”

тема “**Багатосторінковий веб-сайт**”

|  |  |  |
| --- | --- | --- |
| Виконав  студент II курсу  групи КП-92  Гришаєв Дмитро Ігорович  (*прізвище, ім’я, по батькові*)  варіант №4 |  | Перевірив  “\_\_\_\_” “\_\_\_\_\_\_\_\_\_\_\_\_” 20\_\_\_ р.  викладач  Гадиняк Руслан Анатолійович  (*прізвище, ім’я, по батькові*) |

Київ 2020

**Мета роботи**

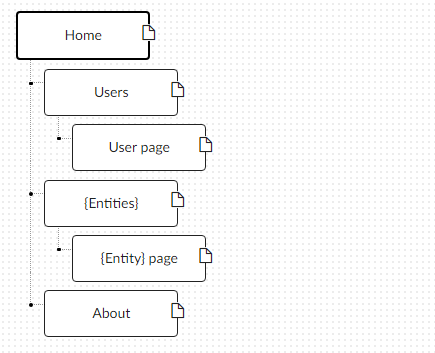
Навчитись створювати веб-сайт, що надає CRUD доступ до ресурсів веб-сервера.

Ознайомитись із шаблонізаторами веб-сторінок для генерації контенту.

Навчитись створювати HTML-форми для взаємодії користувача із веб-сайтом та обробляти на сервері внесені користувачем дані.

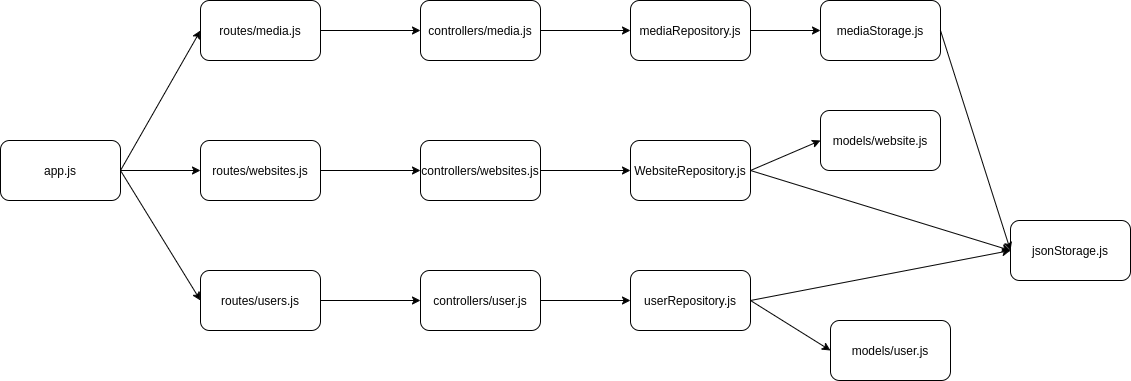
**Постановка завдання**

Реалізувати багатосторінковий веб-сайт із заданою структурою розділів:



Забезпечити можливість пошуку, пагінації, додавання та видалення сутностей за варіантом.

**Діаграма залежності модулів**



**Код всіх всіх нових та змінених модулів**

|  |
| --- |
| **app.js** |
| const express = require('express');  const userRouter = require('./routes/users.js');  const websiteRouter = require('./routes/websites.js');  const mediaRouter = require('./routes/media.js');  const homeRouter = require('./routes/home.js');  const bodyParser = require("body-parser");  const busboy = require('busboy-body-parser');  const mustache = require('mustache-express');  const path = require('path');  const app = express();  const morgan = require('morgan');  const busboyOptions = {  limit: '10mb',  multi: false,  }  const viewDir = path.join(\_\_dirname, 'views');  const partialsDir = path.join(viewDir, 'partials');  app.engine('mst', mustache(partialsDir));  app.set('views', viewDir);  app.set('view engine', 'mst');  app.use(morgan('dev'));  app.use(express.static('public'));  app.use(bodyParser.urlencoded({ extended: true }));  app.use(bodyParser.json());  app.use(busboy(busboyOptions));  app.use('/api/media', mediaRouter);  app.get('/about', (req, res) => {  res.render('about');  });  app.get('/', (req, res) => {  res.render('index');  });  app.use('/users', userRouter);  app.use('/websites', websiteRouter);  app.get('/data/media/:url', (req, res) => {  res.sendFile(${\_\_dirname}/data/media/${req.params.url});  });  app.listen(3001, () => console.log('ready')); |

|  |
| --- |
| **websites.js** |
| //models/website.js  /\*\*  \* @typedef Website  \* @property {integer} id  \* @property {string} name.required - unique name  \* @property {string} address.required - unique address  \* @property {integer} visits\_per\_day - amount of visits per day  \* @property {integer} count\_of\_ad\_bunners - amount of bunners on pages  \* @property {date} date\_of\_creating - the date of creating website  \*/  class Website {  constructor(id, name, address, visits\_per\_day, count\_of\_ad\_bunners, date\_of\_creating, description, url) {  this.id = id; // id  this.name = name; // string  this.address = address; // string  this.visits\_per\_day = visits\_per\_day; //number  this.count\_of\_ad\_bunners = count\_of\_ad\_bunners; //number  this.date\_of\_creating = date\_of\_creating; // Date ISO 8601  this.description = description //string  this.url = url;  }  };  module.exports = Website; |

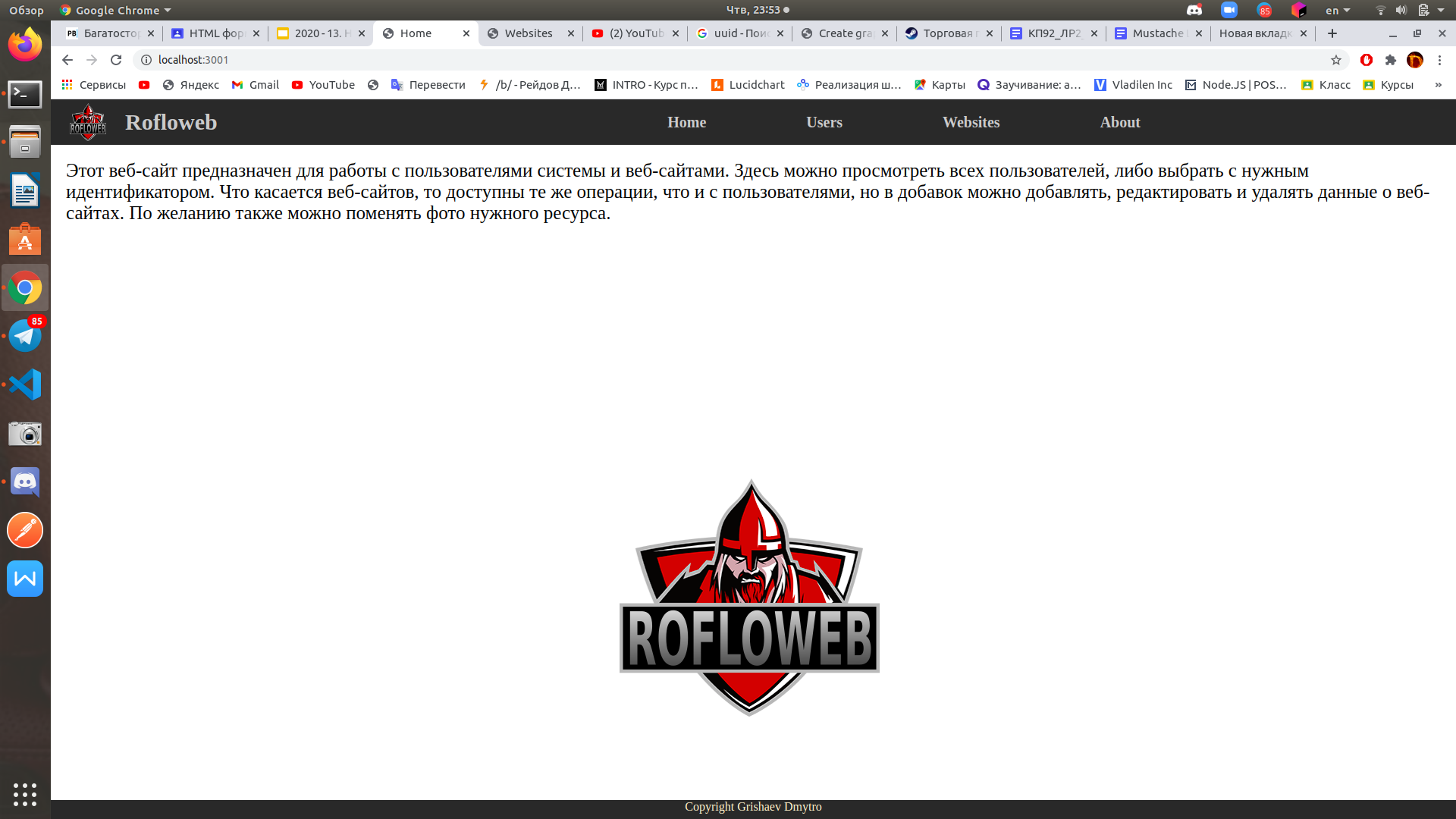
|  |
| --- |
| **user.js** |
| //models/user.js  /\*\*  \* @typedef User  \* @property {integer} id  \* @property {string} login.required - unique username  \* @property {string} fullname - user`s fullname  \* @property {integer} role - user`s role  \* @property {date} registeredAt - the day when user registered in the system  \* @property {string} avaUrl - url of the picture  \* @property {bool} isEnabled - is user enabled  \*/  class User {  constructor(id, login, fullname, role, registeredAt, avaUrl, isEnabled, biography) {  this.id = id; // number  this.login = login; // string  this.fullname = fullname; // string  this.role = role; //number 0 || 1  this.registeredAt = registeredAt; // Date ISO 8601  this.avaUrl = avaUrl; //strign (URL)  this.isEnabled = isEnabled; //bool  this.biography = biography; //bool  }  };  module.exports = User; |

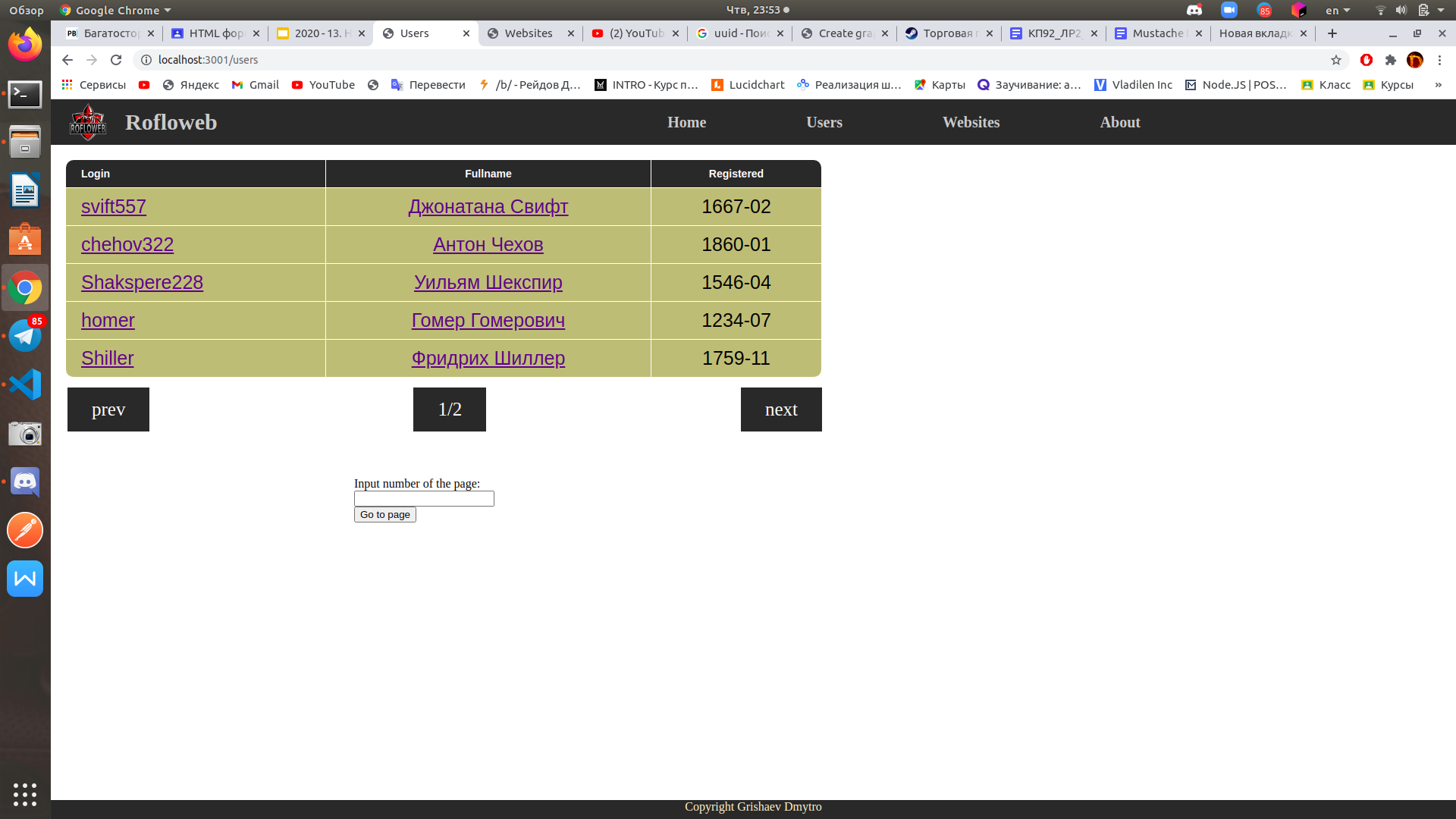
|  |
| --- |
| **routes/websites.js** |
| **const express = require('express');**  **const websiteController = require('../controllers/websites.js');**  **const router = express.Router();**  **/**  **\* Description of getting all websites**  **\* @route GET /api/websites**  **\* @group Websites - website operations**  **\* @param {integer} page.query - page number**  **\* @param {integer} per\_page.query - items per page**  **\* @returns {Array.<Website>} 200 - a page with websites**  **\*/**  **/**  **\* Description of getting website by id**  **\* @route GET /api/websites/{id}**  **\* @group Websites - website operations**  **\* @param {integer} id.path.required - id of the Website**  **\* @returns {Website.model} 200 - Website object**  **\* @returns {Error} 404 - Website not found**  **\*/**  **/**  **\* Description of getting all websites**  **\* @route POST /api/websites**  **\* @group Websites - website operations**  **\* @param {Website.model} id.body.required - new Website object**  **\* @returns {Website.model} 201 - added Website object**  **\* @returns {Error} 400 - Website wasnt added**  **\*/**  **/**  **\* Description of getting all websites**  **\* @route PUT /api/websites**  **\* @group Websites - website operations**  **\* @param {Website.model} id.body.required - new Website object**  **\* @returns {Website.model} 200 - changed Website object**  **\* @returns {Error} 404 - Website wasnt updated**  **\*/**  **/\*\***  **\* Description of getting website by id**  **\* @route DELETE /api/websites/{id}**  **\* @group Websites - website operations**  **\* @param {integer} id.path.required - id of the Website**  **\* @returns {Website.model} 200 - deleted Website object**  **\* @returns {Error} 404 - Website not found**  **\*/**  **router**  **.get("/new", websiteController.createWebsite)**  **.get("/", websiteController.getWebsites)**  **.get("/:id", websiteController.getWebsitesById)**  **.post("/", websiteController.addWebsite)**  **.put("/", websiteController.updateWebsite)**  **.post("/:id", websiteController.deleteWebsites)**  **module.exports = router;** |

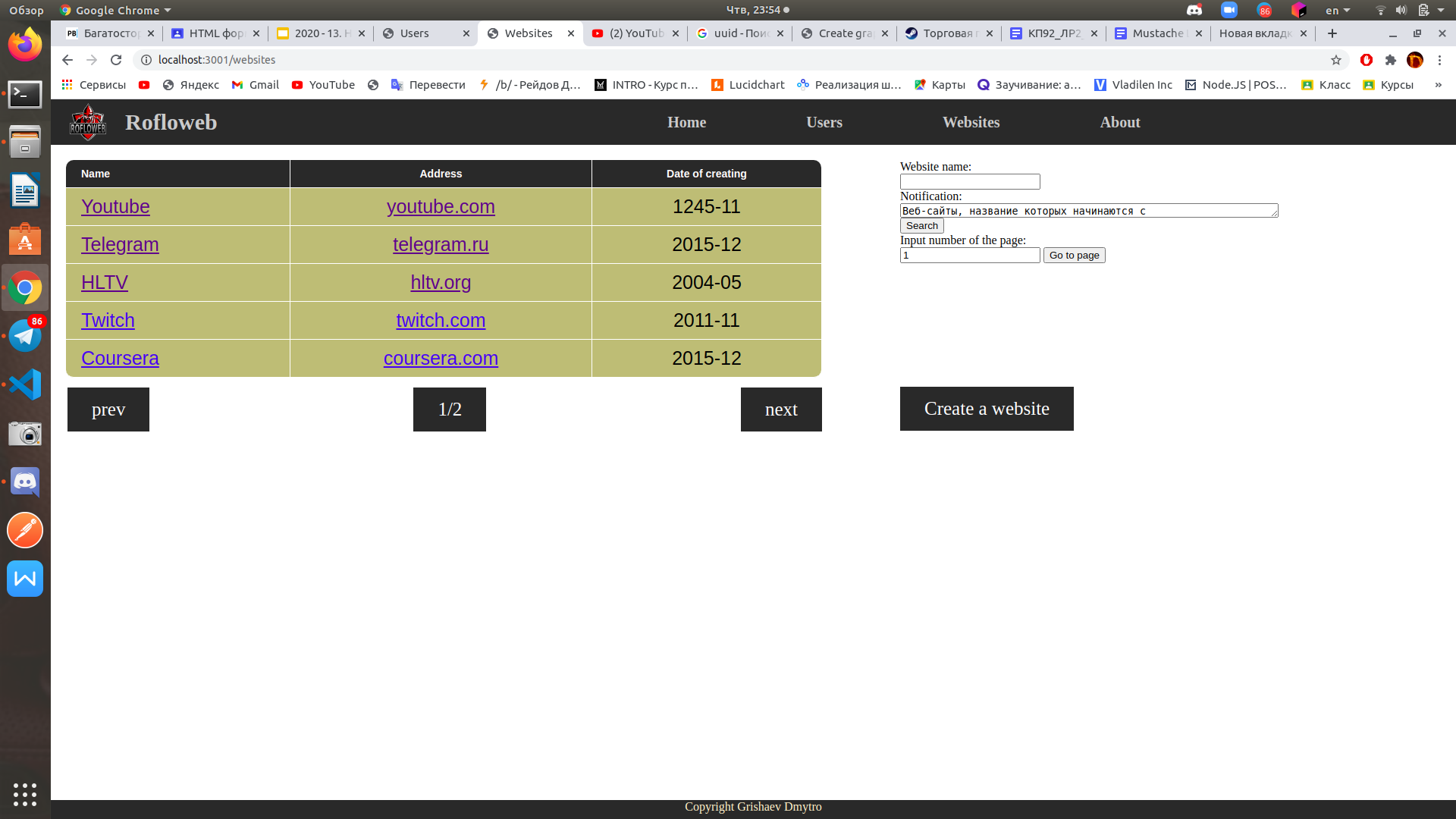
|  |
| --- |
| **controllers/users.js** |
| **const user\_rep = require('../repositories/userRepository.js');**  **const { use } = require('../routes/users.js');**  **const userRepository = new user\_rep("data/users.json");**  **module.exports = {**  **getUsers(req, res) {**  **let page = Number.parseInt(req.param("page") || 1);**  **let records = Number.parseInt(req.param("per\_page") || 5);**  **if (!Number.isInteger(page) || !Number.isInteger(records)) {**  **res.statusCode = 404;**  **let arr = []**  **res.render('users', { arr });**  **} else {**  **res.statusCode = 200;**  **let arr = userRepository.getUsers();**  **let record\_count = arr.length;**  **let page\_count = Math.ceil(record\_count / records);**  **let left = records \* (page - 1);**  **let right = records \* page**  **let current = page;**  **let next = page + 1;**  **let prev = page - 1;**  **let currnall = ${page}/${page\_count};**  **arr = arr.slice(left, right);**  **if (page < 1) {**  **arr = [];**  **prev = 0;**  **next = 1;**  **res.render('users', { arr, current, next, prev, currnall });**  **} else if (page > page\_count) {**  **arr = [];**  **next = page\_count + 1;**  **prev = 2;**  **res.render('users', { arr, current, next, prev, currnall });**  **} else {**  **res.render('users', { arr, current, next, prev, currnall });**  **}**  **}**  **},**  **getUserById(req, res) {**  **const user = userRepository.getUserById(req.params.id);**  **res.statusCode = 200;**  **res.render('user', { user });**  **}**  **};** |

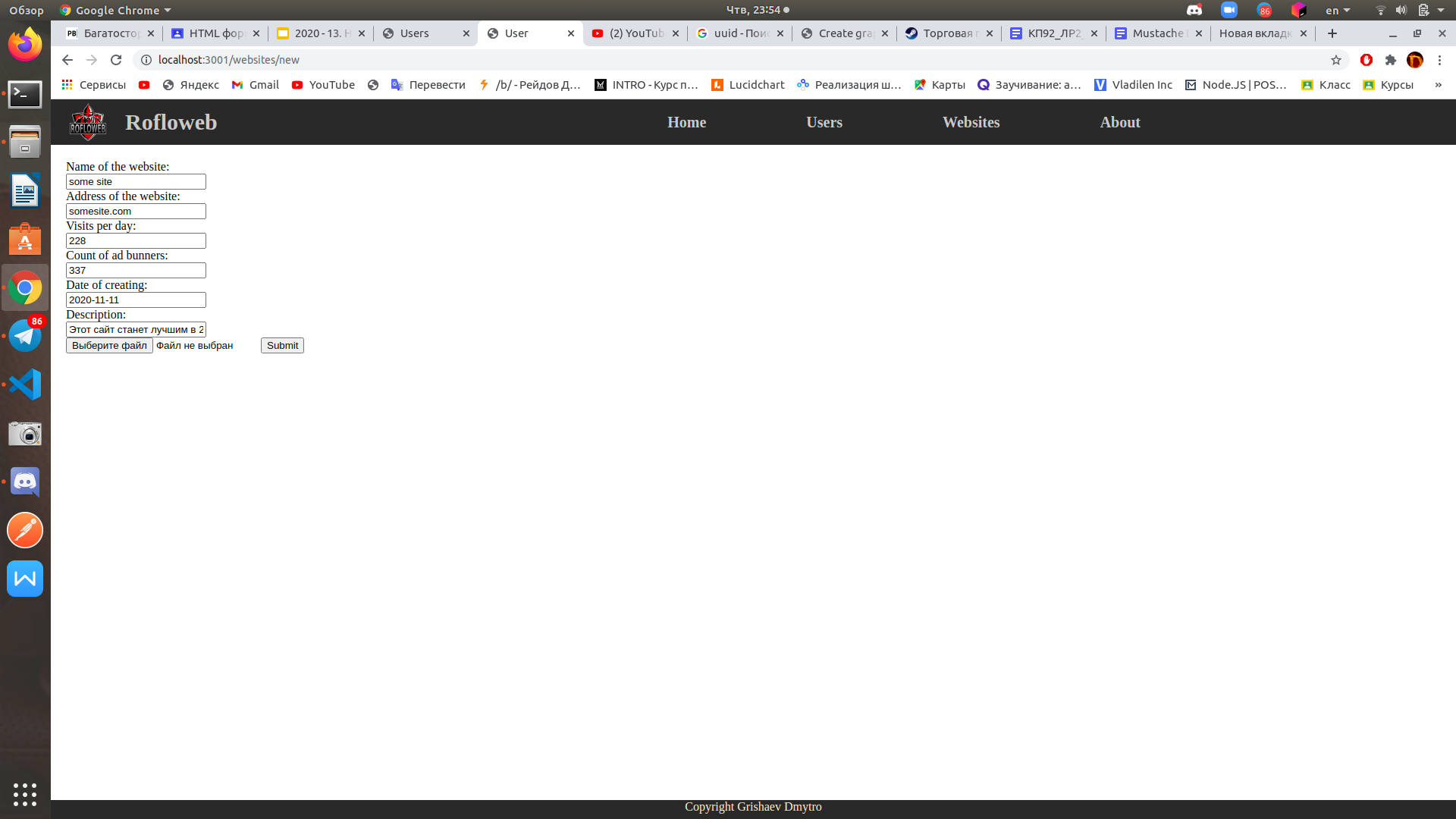
|  |
| --- |
| **controllers/websites.js** |
| **const site\_rep = require('../repositories/WebsiteRepository.js');**  **const media\_rep = require('../repositories/mediaRepository.js');**  **const e = require('express');**  **const websiteRepository = new site\_rep("./data/Websites.json");**  **const mediaRepository = new media\_rep();**  **module.exports = {**  **getWebsites(req, res) {**  **let page = Number.parseInt(req.param("page") || 1);**  **let records = Number.parseInt(req.param("per\_page") || 5);**  **let name = req.param("name") || "";**  **if (!Number.isInteger(page) || !Number.isInteger(records)) {**  **res.statusCode = 404;**  **let arr = []**  **res.render('websites', { arr });**  **} else {**  **res.statusCode = 200;**  **let arr = websiteRepository.getWebsites();**  **if (name !== undefined) {**  **arr = arr.filter(el => el.name.startsWith(name));**  **}**  **let filtered = [];**  **filtered = arr.concat(filtered);**  **let record\_count = arr.length;**  **let page\_count = Math.ceil(record\_count / records);**  **let left = records \* (page - 1);**  **let right = records \* page**  **let current = page;**  **let next = page + 1;**  **let prev = page - 1;**  **let currnall = ${page}/${page\_count};**  **arr = arr.slice(left, right);**  **let str = '';**  **let start = name;**  **if (filtered.length === 0) {**  **str = Веб-сайты, название которых начинаются с ${name}, не найдены;**  **} else {**  **str = Веб-сайты, название которых начинаются с ${name};**  **}**  **if (page < 1) {**  **arr = [];**  **prev = 0;**  **next = 1;**  **res.render('websites', { arr, str, current, next, prev, currnall, start, name });**  **} else if (page > page\_count) {**  **arr = [];**  **next = page\_count + 1;**  **prev = page\_count;**  **res.render('websites', { arr, str, current, next, prev, currnall, start, name });**  **} else {**  **res.render('websites', { arr, str, current, next, prev, currnall, start, name });**  **}**  **}**  **},**  **getWebsitesById(req, res) {**  **res.statusCode = 200;**  **const website = websiteRepository.getWebsitesById(req.params.id)**  **res.render('website', { website });**  **},**  **addWebsite(req, res) {**  **if (req.body.name === undefined || req.body.address === undefined) {**  **res.set("Content-Type", "application/json");**  **res.statusCode = 400;**  **res.send(JSON.stringify('Bad request'));**  **} else {**  **mediaRepository.addMedia(req.files);**  **res.statusCode = 201;**  **let site = req.body;**  **console.log(\_\_dirname);**  **site.url = /data/media/${req.files.photoFile.name};**  **const id = websiteRepository.addWebsite(site);**  **const website = websiteRepository.getWebsitesById(id);**  **res.redirect(/websites/${id});**  **}**  **},**  **updateWebsite(req, res) {**  **res.set("Content-Type", "application/json");**  **const website = websiteRepository.updateWebsite(req.body)**  **if (website === null) {**  **res.statusCode = 404;**  **res.send(Any website wasnt updated);**  **} else if (req.body.name === undefined || req.body.address === undefined) {**  **res.statusCode = 400;**  **res.send(JSON.stringify('Bad request'));**  **} else {**  **res.send(Updated website: ${JSON.stringify(website,null,4)});**  **}**  **},**  **deleteWebsites(req, res) {**  **res.statusCode = 200;**  **const website = websiteRepository.deleteWebsites(req.params.id)**  **res.redirect(/websites);**  **},**  **createWebsite(req, res) {**  **res.render('new');**  **}**  **};** |

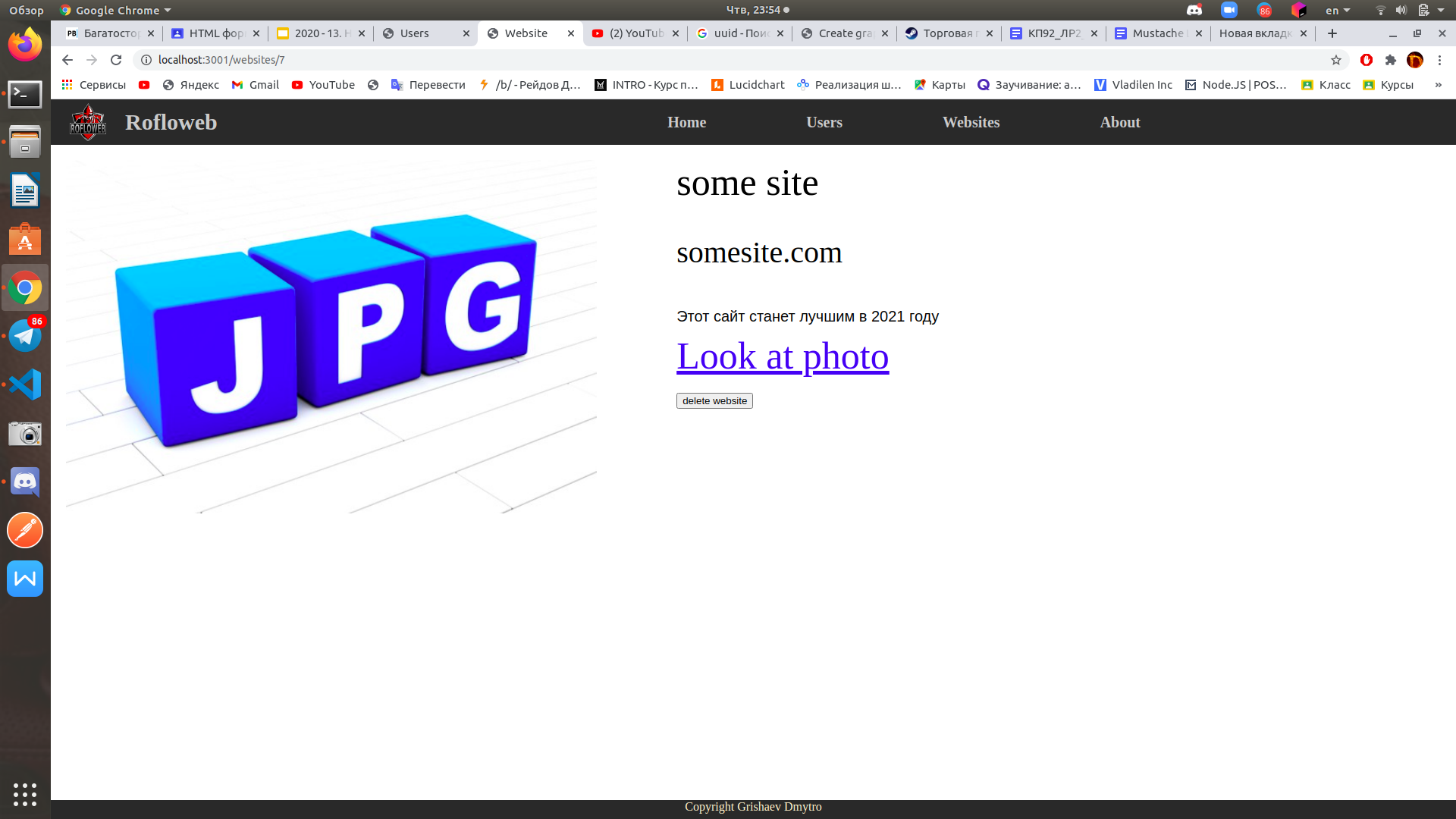
**Зображення всіх розроблених сторінок**















**Висновки**

Виконавши дану лабораторну роботу було навчено створювати веб-сайт, що надає CRUD доступ до ресурсів веб-сервера, створювати HTML-форми для взаємодії користувача із веб-сайтом та обробляти на сервері внесені користувачем дані. Було ознайомлено із шаблонізаторами веб-сторінок для генерації контенту.   
 Лабораторна робота була виконана на мові JavaScript за допомогою середовища виконання Node.js у редакторі VS code.