const express = require('express');

const bodyParser = require('body-parser');

const fs = require('fs').promises;

const path = require('path');

const { read, utils, writeFile } = require('xlsx');

const app = express();

const port = 8000;

// Serve static files from the templates directory

app.use(express.static(path.join(\_\_dirname, 'templates')));

// Set up the body-parser middleware to parse POST request data

app.use(bodyParser.urlencoded({ extended: true }));

app.use(bodyParser.json());

// Load the questions from the q.json file

const questions = require('./q.json');

// Extract the question strings from the questions array

const questionStrings = questions.map(question => question.question);

// Handle POST requests to the /submit route

app.post('/submit', async (req, res) => {

const { name, username, userData } = req.body;

// Validate the data received in the POST request

if (!name || !username || !userData || userData.length !== questions.length) {

return res.status(400).send('Invalid request data');

}

console.log(userData); // Log the value of userData to the console

// Extract the userData values based on the question order in q.json

const userDataValues = questionStrings.map((\_question, index) => userData[index]);

// Create a user directory if it doesn't exist

const userDir = path.join(\_\_dirname, 'users', username);

try {

await fs.mkdir(userDir, { recursive: true }); // Use the recursive option to create the directory if it doesn't exist

} catch (err) {

console.error(err);

return res.status(500).send('Internal server error');

}

// Save the submission to users/results.xlsx

const resultsFile = path.join(\_\_dirname, 'users', 'results.xlsx');

let workbook;

try {

// Check if the file exists

await fs.access(resultsFile);

// If the file already exists, load it

workbook = read(await fs.readFile(resultsFile));

} catch (err) {

if (err.code === 'ENOENT') {

// If the file doesn't exist, create a new workbook and add a header row

workbook = utils.book\_new();

// ...

const headerRow = [

{ header: 'Nume', key: 'name' },

{ header: 'Prenume', key: 'username' },

{ header: 'Varsta', key: 'answer-1' },

{ header: 'Culoare preferata', key: 'answer-2' },

{ header: 'Mancare preferata', key: 'answer-3' },

{ header: 'Locatie', key: 'answer-4' },

{ header: 'Carte preferata', key: 'answer-5' },

...questions.map(question => ({ header: `Q${question.number}`, key: `Q${question.number}` })),

];

utils.book\_append\_sheet(workbook, utils.json\_to\_sheet([], { header: headerRow.map(col => col.header) }), 'Sheet1');

} else {

console.error(err);

return res.status(500).send('Internal server error');

}

}

const worksheet = workbook.Sheets['Sheet1'];

const newSubmission = {

'Nume': name,

'Prenume': username,

'Varsta': userDataValues[0],

'Culoare preferata': userDataValues[1],

'Mancare preferata': userDataValues[2],

'Locatie': userDataValues[3],

'Carte preferata': userDataValues[4],

...questions.reduce((acc, question, index) => {

acc[`Q${question.number}`] = userDataValues[index + 5];

return acc;

}, {})

};

utils.sheet\_add\_json(worksheet, [newSubmission], { skipHeader: true, origin: -1 });

// Write the updated workbook to the file

try {

await writeFile(workbook, resultsFile);

// Send a success response

res.send('Success!');

} catch (error) {

console.error(error);

res.status(500).send('Internal server error');

}

});

// Serve the questions on the /questions route

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

res.send(questions);

});

// Start the server

app.listen(port, () => {

console.log(`Server listening at http://cetateanul-targujiu.com:${port}`);

});