Для реализации RESTful API сервиса заметок в рамках задачи был использован подход с симуляцией доступа к хранилищу данных. В рамках задачи данные хранятся в массиве в оперативной памяти.

При обработке запросов функции-обработчики обращаются к коллекции данных и при необходимости модифицируют ее.

Решение написано на языке JavaScript на платформе node.js с использованием фреймворка express. Обработчики для решения задачи находятся в файле routes/notes.js.

Реализация обработчиков:

const ***express*** = require('express');  
const ***router*** = ***express***.Router();  
const fs = require('fs');  
  
const config = ***JSON***.parse(fs.readFileSync('./notesConfig.json').toString());  
  
const mockNotes = [  
 {  
 id: 1,  
 title: 'oiqweuqoie',  
 content: 'fowhowhqfwefwoqef'  
 },  
 {  
 id: 2,  
 title: 'whfowqeowqe',  
 content: '[ppqwpqwp[wq[qwpwq'  
 },  
 {  
 id: 3,  
 content: 'euoquieouoiujdiwqehjohdijk'  
 }  
];  
  
***router***.get('/', (req, res) => {  
 let notes = [...mockNotes];  
  
 notes = notes.map(n => n.title ? n : {...n, title: n.content  
 .slice(0, config.titlePlaceholderLength < n.content.length ? config.titlePlaceholderLength : n.content.length)});  
  
 if (req.query.query) {  
 notes = notes.filter(n => n.content.includes(req.query.query) || (n.title && n.title.includes(req.query.query)));  
 }  
  
 res.send(notes);  
});  
  
***router***.post('/', (req, res) => {  
 const note = {  
 id: ***Math***.max(...mockNotes.map(n => n.id)) + 1,  
 title: req.body.title,  
 content: req.body.content  
 }  
  
 mockNotes.push(note);  
  
 res.send(note);  
});  
  
***router***.get('/:id', (req, res) => {  
 if (isNaN(+req.params.id)) {  
 res.sendStatus(400);  
 }  
  
 let notes = [...mockNotes];  
  
 notes = notes.map(n => n.title ? n : {n, title: n.content.slice(titlePlaceholderLength)});  
  
 const note = notes.map(n => n.id).indexOf(+req.params.id);  
  
 if (note >= 0) {  
 res.send(notes[note]);  
 } else {  
 res.sendStatus(404);  
 }  
});  
  
***router***.put('/:id', (req, res) => {  
 if (isNaN(+req.params.id)) {  
 res.sendStatus(400);  
 }  
  
 const note = mockNotes.map(n => n.id).indexOf(+req.params.id);  
  
 if (note >= 0) {  
 mockNotes[note].content = req.body.content;  
 mockNotes[note].title = req.body.title;  
 } else {  
 res.sendStatus(404);  
 }  
});  
  
***router***.delete('/:id', (req, res) => {  
 if (isNaN(+req.params.id)) {  
 res.sendStatus(400);  
 }  
  
 const note = mockNotes.map(n => n.id).indexOf(+req.params.id);  
  
 if (note >= 0) {  
 mockNotes.splice(note, 1);  
 } else {  
 res.sendStatus(404);  
 }  
});  
  
module.exports = ***router***;