// components/toolbar.vue

<template>

<div>

<v-app-bar

style="-webkit-app-region: drag"

color="primary"

dark

>

<v-btn

icon

style="-webkit-app-region: no-drag;"

@click="$router.push('/')"

v-if="$route.name !== 'welcome'">

<v-icon>mdi-arrow-left</v-icon>

</v-btn>

<v-toolbar-title>

Сканер QR-кодов

</v-toolbar-title>

<v-spacer></v-spacer>

<v-btn icon @click="minimizeApp" style="-webkit-app-region: no-drag;"

>

<v-icon>mdi-window-minimize</v-icon>

</v-btn>

<v-btn icon @click="closeApp" style="-webkit-app-region: no-drag;"

>

<v-icon>mdi-close</v-icon>

</v-btn>

</v-app-bar>

</div>

</template>

<script>

import {remote} from "electron";

export default {

name: 'Toolbar',

mounted() {

},

methods: {

closeApp() {

const window = remote.getCurrentWindow();

window.close();

},

minimizeApp() {

const window = remote.getCurrentWindow();

window.minimize();

}

}

}

</script>

//router/index.js

import Vue from 'vue'

import VueRouter from 'vue-router'

import Welcome from '@/views/Welcome';

import WorkSpace from '@/views/Workspace';

Vue.use(VueRouter);

const routes = [

{

name: 'welcome',

path: '/',

component: Welcome

},

{

name: 'workspace',

path: '/workspace',

component: WorkSpace

}

];

const router = new VueRouter({

mode: 'hash',

base: '/',

routes

})

export default router;

//views/Welcome.vue

<template>

<div class="d-flex justify-center align-center">

<v-btn color="primary" @click="navToWorkspace">Начать сканирование</v-btn>

</div>

</template>

<script>

export default {

methods: {

navToWorkspace() {

this.$router.push('workspace'); } } }

</script>

<style scoped>

div {

height: calc(100vh - 64px);

}

</style>

// views/Workspace.vue

<template>

<v-card class="p-1 h-100">

<v-dialog v-model="confirmDialog" max-width="500" persistent="">

<v-card>

<v-card-title

class="headline primary"

primary-title

>

<span class="white--text">Подтвердите действие</span>

</v-card-title>

<v-card-text style="padding: 30px;">

<h2>Хотите продолжить сканирование?</h2>

</v-card-text>

<v-divider></v-divider>

<v-card-actions>

<v-spacer/>

<v-btn

color="error"

text

@click="$router.push('/')"

>

Нет

</v-btn>

<v-btn

color="success"

text

@click="confirmDialog = false; loadingSpinner = false;"

>

Да

</v-btn>

</v-card-actions>

</v-card>

</v-dialog>

<v-dialog v-model="dialog" max-width="600" persistent>

<v-card>

<div style="

padding: 20px;

">

<v-progress-circular

indeterminate

color="primary"

></v-progress-circular>

<span style="margin-left: 20px;">Идет сканирование...</span>

</div>

</v-card>

</v-dialog>

<v-container fluid>

<div v-if="loadingSpinner" class="spinner-container d-flex justify-center align-center h-100">

<v-progress-circular indeterminate color="primary"></v-progress-circular>

</div>

<v-row class="h-100">

<v-col style="position: relative;" v-if="!loadingSpinner" sm="8">

<div v-if="scans.length">

<v-btn color="error" class="mb-2" @click="clearData" style="width: 215px;">

Очистить список

<v-icon>mdi-delete</v-icon>

</v-btn>

<v-btn color="success" class="mb-2 ml-5" style="width: 215px;" @click="importExcel">

Экспорт

<v-icon>mdi-upload</v-icon>

</v-btn>

<h4>Всего объектов: {{ scans.length }}</h4>

<v-simple-table v-slot:default style="max-height: 60vh; overflow-y: scroll;" id="main\_table">

<thead>

<tr>

<th class="text-left w-5">#</th>

<th class="text-left">ФИО/Наименование</th>

<th class="text-left">ИИН/БИН</th>

<th class="text-left w-50">Действие</th>

</tr>

</thead>

<tbody>

<tr v-for="(scan, index) of scans" :key="index">

<td>{{ index + 1 }}</td>

<td>

<span v-if="index !== editableIndex">

{{ scan.name }}

</span>

<v-text-field v-else v-model="scan.name"/>

</td>

<td>

<span v-if="index !== editableIndex">

{{ scan.IIN }}

</span>

<v-text-field v-else v-model="scan.IIN"/>

</td>

<td>

<div v-if="editableIndex !== index" class="d-flex">

<v-btn icon @click="deleteScan(index)">

<v-icon>mdi-delete</v-icon>

</v-btn>

<v-btn icon @click="editScan(index)">

<v-icon>mdi-pencil</v-icon>

</v-btn>

</div>

<v-btn v-else color="success" icon>

<v-btn icon @click="editableIndex = null">

<v-icon>mdi-check</v-icon>

</v-btn>

</v-btn>

</td>

</tr>

</tbody>

</v-simple-table>

</div>

<div v-else

style="max-width: 300px; position: absolute; top: 50%; left: 50%; transform: translate(-50%, -50%);">

<h4 style="text-align: center;">Нет данных!</h4>

<p style="text-align: center;">Для начала сканирования поднесите документ с QR-кодом к

веб-камере</p>

</div>

</v-col>

<v-col class="d-flex justify-center align-center" sm="4">

<video ref="video" v-show="!loadingSpinner"/>

<canvas ref="canvas"/>

</v-col>

</v-row>

</v-container>

</v-card>

</template>

<script>

import jsQr from "jsqr";

import {ipcRenderer, remote} from 'electron';

import XLSX from 'xlsx';

import {saveAs} from 'file-saver';

export default {

name: "WorkSpace",

data: () => ({

scans: [],

dialog: false,

editableIndex: null,

urls: [],

fullText: '',

loadingSpinner: true,

confirmDialog: false

}),

async mounted() {

await this.loadWebcam();

await this.parseData();

},

watch: {

scans: function (oldScans, newScans) {

const table = document.querySelector('#main\_table');

setTimeout(() => {

table.scrollTop = table.scrollHeight;

}, 0)

}

},

methods: {

async loadWebcam() {

let stream = null;

stream = await navigator.mediaDevices.getUserMedia({

audio: false,

video: true

});

const video = this.$refs.video;

video.srcObject = stream;

video.onloadedmetadata = () => {

this.loadingSpinner = false;

video.play();

requestAnimationFrame(this.tick);

};

},

playAudio() {

const audio = new Audio(require("../../assets/beep.wav"));

audio.play();

},

async tick() {

const tick = () => {

const video = document.querySelector("video");

const canvasElement = document.querySelector("canvas");

canvasElement.hidden = true;

const canvas = canvasElement.getContext("2d");

const width = video.videoWidth;

const height = video.videoHeight;

canvasElement.height = height;

canvasElement.width = width;

canvas.drawImage(video, 0, 0, width, height);

const imageData = canvas.getImageData(0, 0, width, height);

const code = jsQr(imageData.data, imageData.width, imageData.height, {

inversionAttempts: "dontInvert"

});

if (code && code.data.length) {

let url = code.data;

if (!this.validURL(url)) {

requestAnimationFrame(tick);

return;

}

if (!this.isDuplicate(url)) {

this.urls.push(url);

this.downloadFile(url);

}

}

requestAnimationFrame(tick);

};

tick();

},

async stopScanning() {

const canvasElement = document.querySelector("canvas");

const ctx = canvasElement.getContext("2d");

ctx.clearRect(0, 0, canvasElement.width, canvasElement.height);

},

isDuplicate(scan) {

return this.urls.includes(scan);

},

deleteScan(index) {

this.scans.splice(index, 1);

},

clearData() {

this.scans = [];

},

validURL(str) {

var pattern = new RegExp('^(https?:\\/\\/)?' +

'((([a-z\\d]([a-z\\d-]\*[a-z\\d])\*)\\.)+[a-z]{2,}|' +

'((\\d{1,3}\\.){3}\\d{1,3}))' +

'(\\:\\d+)?(\\/[-a-z\\d%\_.~+]\*)\*' +

'(\\?[;&a-z\\d%\_.~+=-]\*)?' +

'(\\#[-a-z\\d\_]\*)?$', 'i');

return !!pattern.test(str);

},

downloadFile(url) {

this.dialog = true;

ipcRenderer.send('download-pdf', {

url: url

})

},

editScan(index) {

this.editableIndex = index;

},

parseData() {

const nameRegex = /в отношении должника(.\*)(БИН|ИИН|и наложить арест)/;

ipcRenderer.on('file-parsed', (e, data) => {

let fullText = [];

data.pages.forEach(page => {

const text = page.content.reduce((a, c) => {

return a + c.str + ' ';

}, "");

fullText.push(text);

});

this.fullText = fullText.join('');

let plainText = /в отношении должника(.\*)(БИН|ИИН|)и наложить арест/.exec(this.fullText)[0];

let name = nameRegex.exec(plainText);

name = name[1].trim();

let found = name.match(/(.\*)(ИИН|БИН)(.\*)/);

if (!found) {

this.scans.push({

name: name,

IIN: 'Не указан'

})

}

else {

this.scans.push({

name: found[1].trim(),

IIN: found[3].trim()

})

}

this.dialog = false;

this.playAudio();

});

},

async importExcel() {

this.loadingSpinner = true;

const wb = XLSX.readFile('assets/template.xlsx', {cellStyles: true});

const outputArray = [];

const today = new Date();

const date = `${today.getDate()}.${today.getMonth() + 1}.${today.getFullYear()}`;

this.scans.forEach((s, i) => {

outputArray.push({

'№': i + 1,

'ФИО/Наименование': s.name,

'ИИН/БИН': s.IIN,

});

});

XLSX.utils.sheet\_add\_json(wb.Sheets[wb.SheetNames[0]], outputArray);

const wbout = XLSX.write(wb, {type: 'array', bookType: 'xlsx'});

await saveAs(new Blob([wbout], {type: 'application/octet-stream'}), `Список от ${date}.xlsx`);

this.confirmDialog = true

}

}

};

</script>

<style lang="scss">

.w-50 {

width: 100px;

}

.spinner-container {

height: 600px;

}

.h-100 {

height: calc(100vh - 64px);

}

.vh-50 {

height: 50vh;

}

video {

width: 400px;

height: 300px;

}

.p-3 {

padding: 30px;

}

</style>

//App.vue

<template>

<v-app app>

<Toolbar/>

<router-view/>

</v-app>

</template>

<script>

import Toolbar from "./components/toolbar";

export default {

components: {

Toolbar

},

};

</script>

<style>

#app {

-webkit-user-select: none;

}

</style>

// background.js

'use strict'

import path from 'path';

import fs from 'fs';

import rimraf from 'rimraf';

import {

app,

protocol,

BrowserWindow,

ipcMain,

ipcRenderer

} from 'electron'

import {

createProtocol,

installVueDevtools

} from 'vue-cli-plugin-electron-builder/lib'

import DownloadManager from 'electron-download-manager';

DownloadManager.register({

downloadFolder: app.getPath('downloads') + '/qr-scanner'

});

const PDFExtract = require('pdf.js-extract').PDFExtract;

const pdfExtract = new PDFExtract();

const isDevelopment = process.env.NODE\_ENV !== 'production';

let win;

protocol.registerSchemesAsPrivileged([{

scheme: 'app',

privileges: {

secure: true,

standard: true

}

}])

function createWindow() {

win = new BrowserWindow({

width: 1280,

height: 720,

frame: false,

icon: app.getAppPath() + '/icon.png',

webPreferences: {

nodeIntegration: true,

devTools: true,

}

})

if (process.env.WEBPACK\_DEV\_SERVER\_URL) {

win.loadURL(process.env.WEBPACK\_DEV\_SERVER\_URL)

if (!process.env.IS\_TEST) win.webContents.openDevTools()

} else {

createProtocol('app');

win.loadURL('app://./index.html')

}

win.on('closed', () => {

win = null

})

}

app.on('window-all-closed', () => {

if (process.platform !== 'darwin') {

app.quit()

}

})

app.on('activate', () => {

if (win === null) {

createWindow()

}

})

app.on('ready', async () => {

createWindow();

app.commandLine.appendSwitch('inspect', '5858');

ipcMain.on('download-pdf', async (event, info) => {

DownloadManager.download({

url: info.url

}, function (err, data) {

console.log(data);

console.log(err);

const path = data.filePath;

pdfExtract.extract(path, {}, async (err, data) => {

if (err) return console.log(err);

const currrentWin = BrowserWindow.getFocusedWindow();

currrentWin.webContents.send('file-parsed', data);

});

});

});

});

app.on('before-quit', async () => {

const directory = app.getPath('downloads') + '/qr-scanner';

deleteFolderRecursive(directory);

});

const deleteFolderRecursive = function (path) {

if (fs.existsSync(path)) {

fs.readdirSync(path).forEach(function (file) {

var curPath = path + "/" + file;

if (fs.lstatSync(curPath).isDirectory()) {

deleteFolderRecursive(curPath);

} else {

fs.unlinkSync(curPath);

}

});

}

};

if (isDevelopment) {

if (process.platform === 'win32') {

process.on('message', data => {

if (data === 'graceful-exit') {

app.quit()

}

})

} else {

process.on('SIGTERM', () => {

app.quit()

})

}

}

// main.js

import Vue from 'vue'

import App from './App.vue'

import router from './router'

import vuetify from './plugins/vuetify';

Vue.config.productionTip = false;

new Vue({

router,

vuetify,

render: h => h(App)

}).$mount('#app');