A screenshot of a computer

AI-generated content may be incorrect.

Public class GuiForm {

Private JPanel pnlMain

Private JFrame form;

Public GuiForm() {

Form = new JFrame(“cime”)

Form.setSize(320, 240)

Form.setDefaultCloseOperation(WindowConstants.EXIT\_ON\_CLOSE);

Form.setContenmtPane(pnlMain);

JMenuItem mnuKilepes = new JMenuItem(‘kilepes’)

JMenuItem mnuBetoltes = new JMenuItem(‘betoltes’)

JMenu mnu = new JMenu(‘program’)

Mnu.add(mnuBetoltes);

Mnu.add(new JSeparator());

Mnu.add(mnuKilepes);

JMenuBar mb = new JMenubar();

Mb.add(mnu);

Form.setJMenuBar(mb);

mnuBetoltes.AddActionListener( {

@Override

Public void actionPerformed(ActionEvent e){

JFileChooser fc = new JFIleChooser()’

Int gomb = fc.showOpenDialog(null);

//mit fog csinalni ez a menu

Try {List<String> sorok = Files.readAllLines(Path.of(fajl neve))

} catch (IOException ex){

System.err.println(‘I/O hiba’)

Throw new RuntimeException(ex);

}

}

});

Form.setVisible(true);

}

}

**🔧 1. Backend (Laravel) – 15 pont**

**✅ 1.1. Projekt létrehozása (1 pont)**

bash

CopyEdit

composer create-project laravel/laravel receptek-api

cd receptek-api

php artisan serve

**Első commit neve:**  
"init commit – Laravel REST API setup"

**✅ 1.2. Kategória tábla létrehozása (2 pont)**

bash

CopyEdit

php artisan make:model Kategoria -m

database/migrations/xxxx\_xx\_xx\_create\_kategorias\_table.php:

php

CopyEdit

public function up()

{

Schema::create('kategorias', function (Blueprint $table) {

$table->id();

$table->string('nev');

$table->timestamps();

});

}

bash

CopyEdit

php artisan migrate

**✅ 1.3. Receptek tábla + kapcsolat (2 pont)**

bash

CopyEdit

php artisan make:model Recept -m

database/migrations/...create\_recepts\_table.php:

php

CopyEdit

public function up()

{

Schema::create('recepts', function (Blueprint $table) {

$table->id();

$table->string('nev');

$table->text('leiras');

$table->foreignId('kategoria\_id')->constrained('kategorias');

$table->timestamps();

});

}

bash

CopyEdit

php artisan migrate

**✅ 1.4. Feltöltés adatbázisba (2 pont)**

bash

CopyEdit

php artisan make:seeder KategoriaSeeder

php artisan make:seeder ReceptSeeder

database/seeders/KategoriaSeeder.php:

php

CopyEdit

Kategoria::create(['nev' => 'Leves']);

Kategoria::create(['nev' => 'Főétel']);

database/seeders/ReceptSeeder.php:

php

CopyEdit

Recept::create([

'nev' => 'Gulyásleves',

'leiras' => 'Finom magyar leves',

'kategoria\_id' => 1,

]);

bash

CopyEdit

php artisan db:seed --class=KategoriaSeeder

php artisan db:seed --class=ReceptSeeder

**✅ 1.5. Recept lekérdezés kategória nevével (2 pont)**

**Modellekben kapcsolat:**

**Kategoria.php**

php

CopyEdit

public function receptek() {

return $this->hasMany(Recept::class);

}

**Recept.php**

php

CopyEdit

public function kategoria() {

return $this->belongsTo(Kategoria::class);

}

**✅ 1.6. Routing (2 pont)**

routes/api.php:

php

CopyEdit

Route::get('/receptek', [ReceptekController::class, 'index']);

Route::get('/receptek/szures/{id}', [ReceptekController::class, 'szures']);

Route::get('/kategoriak', [KategoriaController::class, 'index']);

**✅ 1.7. Controllerek (2 pont)**

bash

CopyEdit

php artisan make:controller ReceptekController

php artisan make:controller KategoriaController

**ReceptekController.php**

php

CopyEdit

public function index() {

return Recept::with('kategoria')->get();

}

public function szures($id) {

return Recept::with('kategoria')->where('kategoria\_id', $id)->get();

}

**KategoriaController.php**

php

CopyEdit

public function index() {

return Kategoria::all();

}

**✅ 1.8. Adatbázis export (1 pont)**

bash

CopyEdit

mysqldump -u root -p receptek > receptek.sql

**Commit neve:**  
git commit -m "adatbazisexport"

**🎨 2. Frontend (React) – 15 pont**

**✅ 2.1. Projekt létrehozása**

bash

CopyEdit

npx create-react-app receptek-frontend

cd receptek-frontend

npm install axios

**✅ 2.2. Axios beállítása (1 pont)**

src/api.js:

js

CopyEdit

import axios from 'axios';

const api = axios.create({

baseURL: 'http://localhost:8000/api',

});

export default api;

**✅ 2.3. Komponensek (2 pont)**

Hozz létre három komponenst:

* TablazatReceptek.jsx
* SorRecept.jsx
* KategoriaSzuro.jsx

**✅ 2.4. Context és state (2 pont)**

src/context/ReceptContext.jsx:

js

CopyEdit

import { createContext, useState, useEffect } from 'react';

import api from '../api';

export const ReceptContext = createContext();

export const ReceptProvider = ({ children }) => {

const [receptek, setReceptek] = useState([]);

const [kategoriak, setKategoriak] = useState([]);

const [kivalasztottRecept, setKivalasztottRecept] = useState(null);

useEffect(() => {

api.get('/receptek').then(res => setReceptek(res.data));

api.get('/kategoriak').then(res => setKategoriak(res.data));

}, []);

return (

<ReceptContext.Provider value={{

receptek,

kategoriak,

kivalasztottRecept,

setKivalasztottRecept,

setReceptek

}}>

{children}

</ReceptContext.Provider>

);

};

**✅ 2.5. Recept és kategória megjelenítés (2 pont)**

TablazatReceptek.jsx például:

js

CopyEdit

import { useContext } from 'react';

import { ReceptContext } from '../context/ReceptContext';

function TablazatReceptek() {

const { receptek, setKivalasztottRecept } = useContext(ReceptContext);

return (

<table>

<thead>

<tr><th>Név</th><th>Kategória</th></tr>

</thead>

<tbody>

{receptek.map(recept => (

<tr key={recept.id} onClick={() => setKivalasztottRecept(recept)}>

<td>{recept.nev}</td>

<td>{recept.kategoria.nev}</td>

</tr>

))}

</tbody>

</table>

);

}

**✅ 2.6. Kiválasztott recept megjelenítése (2 pont)**

KivalasztottRecept.jsx:

js

CopyEdit

import { useContext } from 'react';

import { ReceptContext } from '../context/ReceptContext';

function KivalasztottRecept() {

const { kivalasztottRecept } = useContext(ReceptContext);

if (!kivalasztottRecept) return null;

return (

<div>

<h2>{kivalasztottRecept.nev}</h2>

<p>{kivalasztottRecept.leiras}</p>

</div>

);

}

**✅ 2.7. Kategória szerinti szűrés (2 pont)**

KategoriaSzuro.jsx:

js

CopyEdit

import { useContext } from 'react';

import { ReceptContext } from '../context/ReceptContext';

import api from '../api';

function KategoriaSzuro() {

const { kategoriak, setReceptek } = useContext(ReceptContext);

const handleClick = (id) => {

api.get(`/receptek/szures/${id}`).then(res => setReceptek(res.data));

};

return (

<div>

{kategoriak.map(k => (

<button key={k.id} onClick={() => handleClick(k.id)}>{k.nev}</button>

))}

</div>

);

}

**✅ 2.8. App komponens összerakása**

App.jsx:

js

CopyEdit

import { ReceptProvider } from './context/ReceptContext';

import TablazatReceptek from './components/TablazatReceptek';

import KategoriaSzuro from './components/KategoriaSzuro';

import KivalasztottRecept from './components/KivalasztottRecept';

function App() {

return (

<ReceptProvider>

<KategoriaSzuro />

<TablazatReceptek />

<KivalasztottRecept />

</ReceptProvider>

);

}

export default App;