

# Receptek

# bevezetés

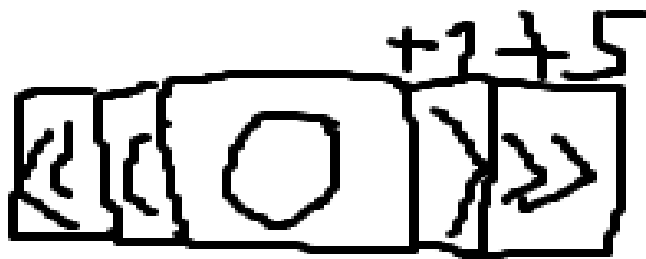
- Programunk egy adatbázis körül épített weboldal, ahol lehetősége van felhasználóknak recepteket megnézni, feltölteni, szerkeszteni, és törölni.
- Azért választottuk ezt a témát, mert főzés szívünkhöz nagyon közel áll, és ezen keresztül tudásunk minden prospektusát be bírjuk mutatni, mint az adatbázis tervezés és kezelés, javascript, css, stb...
- Frontend 40%, Backend 60%

# A program célja

- Segítség a receptek tárolásában
  - Feltöltés
  - Szerkesztés
  - Törlés
- Receptek könnyű megtalálása:
  - mint pl. Kategóriák, A-Z rendezés

# Tervezés

- Paintben történt meg



home

add recipe

search...

recipe name here

image

desc

course type

cook time/prep

instruction/ingredient

edit

delete

# Program funkciói

- Receptek feltöltése
- Recept törlése
- Recept szerkesztése
- Receptek keresése

# A programról

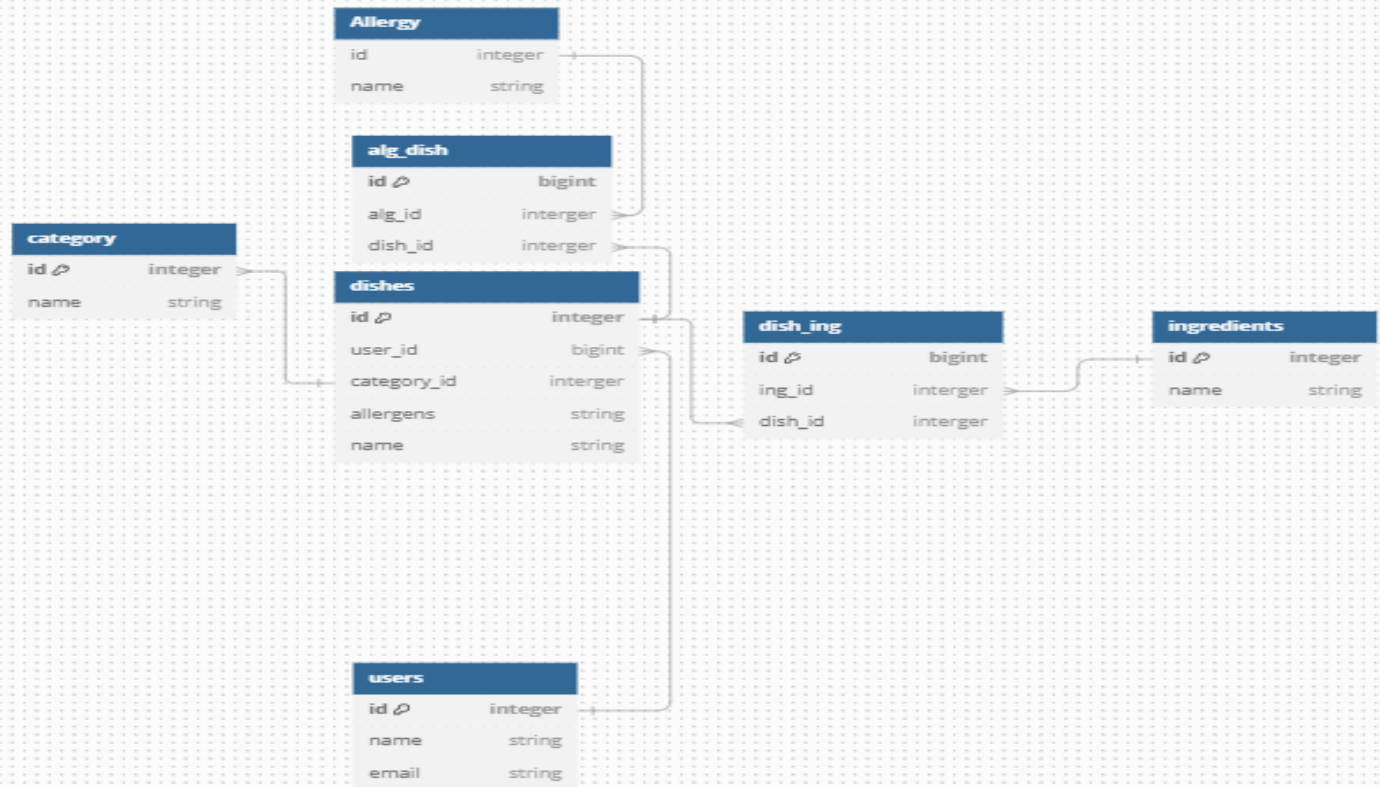
- Backend – Laravel Blade
- Frontend – Laravel blade
- Adatbázis:
  - Mysql

# Backend

- A backend részét a laravel blade keretrendszerének a **DishController** előre generált function-eivel használva írtuk meg. Ezek a web.php-ban vannak meghívva.

# Adatbázis

- Az adatbázisokhoz MySql rendszert használtunk.
- Kezdeti vázlatunk





# Adatbázis

- **A Dish táblába** van a legtöbb adat elementve, mint főbb adatok mint név, kép, stb..foreign key a courses táblához, és adattáblák amik átmenetileg vannak ott mint hozzávalók, és idő ami később saját projektként külön táblába lesz átrakva szűrési szempontból.
- create\_dish migration:

```
public function up(): void
{
    Schema::create('dishes', function (Blueprint $table) {
        $table->id();
        $table->foreignId('courseId')->references('id')->on('courses')->default(8)->onDelete('cascade');
        $table->string('name');
        $table->string('desc')->nullable();
        $table->string('img')->nullable()->default('dish.jpg');
        $table->json('ing');
        $table->json('inst');
        $table->integer('prep')->nullable(); //minutes
        $table->integer('cooktime')->nullable(); //minutes
        $table->timestamps();
    });
}
```

# Adatbázis

- Dish model:

```
class Dish extends Model
{
  0 references | 0 overrides
  public function course(): BelongsTo{
    return $this->belongsTo(related: Course::class, foreignKey: "courseId");
  }
  0 references | 0 overrides
  public function allergens(): BelongsToMany{
    return $this->belongsToMany(related: Allergen::class, table: "alg_dish", foreignPivotKey: "dishid", relatedPiv... "alg");
  }
  0 references
  public $table= 'dishes';
  0 references
  protected $casts=[
    'ing'=>'array',

    "inst"=>"array"
  ];
  0 references
  protected $fillable=["courseId", "name", "desc", "ing", "inst", "img", "prep", "cooktime"];
}
```

# Adatbázis

- Dish seeder:

```
public function run(): void
{
    $Dishes=[
        [
            "name"=>"peanut butter sandwich",
            "courseId"=>"8",
            "desc"=>"yummy pbj suitable for every meal",
            "img"=>"dish.jpg",
            "inst"=>array("take 2 slices of bread"," put peanut butt
            "ing"=>array("bread","peanut butter","jelly",),
            "prep"=>5,
            "cooktime"=>null,
        ],
        [
            "name"=>"garlic bread",
            "courseId"=>"1",
            "desc"=>"yummy garlic bread great with pasta",
            "img"=>"dish.jpg",
            "inst"=>array("take a baguette and cut into it","slather
            "ing"=>array("baguette","garlic butter","cheese"),
            "prep"=>5,
            "cooktime"=>10,
        ],
    ];
    foreach($Dishes as $d){
        Dish::create(attributes: $d);
    }
}

$this->call(class: [
    DishSeeder::class, AlgDishSeeder::class
]);
```

# Backend

- Dishcontroller
  - Az **index function** lehívja az összes rekordot a Dish táblából, és átadja tömbként a **front view-be** a **\$dish** változón keresztül.

```
reference | overrides  
public function index()  
{  
    $dish= Dish::all();  
    return view('front', compact('dish'));  
}
```

# Backend

- A **create function** hasonló, csak itt az **Allergens** és a **Courses** tábláknak hívja le az adatait, amit az **\$alg** és **\$course** változóknak adja át amikor visszaadja a **create view-t**.

```
public function create(): View
{
    $alg= Allergen::all();
    $course= Course::all();
    return view(view: 'create', data: ['alg'=>$alg, 'course'=>$course]);
}
```

# Backend

- A **store function** már komplikáltabb, de mindenek előtt fut egy validálás.

```
public function store(Request $request): RedirectResponse
{
    $request->validate(rules: [
        'name' => 'required|min:3',
        'courseId' => 'required',
        'desc' => 'string',
        'img' => 'nullable|mimes:png,jpg,jpeg',
        'inst' => 'array',
        'ing' => 'array',
        'allergens' => 'array',
        'prep' => 'integer',
        'cooktime' => 'nullable|integer',
    ]);
}
```

# Backend

- Ezután megnézi, hogy jött-e a request-ben egy kép.
- Ha igen, akkor a feltöltött képet a public/dishimg mappába menti el, és a fájl nevét a **\$filename** változóban menti el.
- Viszont ha nem akkor a public/placeholder mappában lévő dish.jpg képre hivatkozik.
- Ez a rész egy [youtube videó](#) alapján lett megírva

```
if($request->has(key: 'image')){
    $img =$request->file(key: 'image');
    $ext= $img->getClientOriginalExtension() ;
    $filename=time().'.'.$ext;
    $imgpath='dishimg/';
    $img->move(directory: $imgpath,name: $filename);
}
else{
    $filename='dish.jpg';
}
```

# Backend

- Ezt követően létrehozza a **Dish táblában** szereplő adatokat, a `::create function` segítségével, amit egyúttal a **\$dish** változóba is belerak.

```
$dish= Dish::create(attributes: [  
  'name'=> $request['name'],  
  'courseId'=>$request['courseId'],  
  'desc'=>$request['desc'],  
  'img'=>$filename,  
  'inst'=> $request['instructions'],  
  'ing'=>$request['ingredients'],  
  'prep'=>$request['prep'],  
  'cooktime'=>$request['cooktime']  
]);
```



# Backend

- Végül megnézi, hogy a requestben érkezett allergens tömbben adat, ha nem akkor redirect-el a **show view**-re.
- Viszont hogy ha érkezett adat, akkor egy foreach végigmegy a tömbön, és az **Alg\_dish** model a ::create function-jével minden allergénnek készít egy rekordot, és hozzáköti a már létrehozott rekordhoz a **\$dish** változó **id**-jával.
- Végül redirect-el a **show view**-re.

<u>2</u>	<u>1</u>
2	7
garlic bread	

2	<u>1</u>
2	7
Gluten	

```
if(empty($request['allergens'])){\n    return redirect(to: 'dish/'.$dish->id)->with(key: 'success',value: '');\n}\nelse{\n    foreach ($request['allergens'] as $a) {\n        alg_dish::create(attributes: [\n            'dishid'=>$dish->id,\n            'alg' => $a,\n        ]);\n    }\n}\n\nreturn redirect(to: 'dish/'.$dish->id)->with(key: 'success',value: '');
```

# Backend

- A **show function** hasonló az index function-hoz, viszont itt csak egy recept adatait hívjuk le **id** alapján.
- Mielőtt még bármi máshoz nyulna, megnézi hogy létezik-e rekord ezzel az id-val, ha nem, akkor redirect-el a **front view**-ra.
- Ha viszont van, akkor lehívja az **Allergens**, és a **Courses** táblákból az összes adatot.
- Végül mindezt átadva redirect-el a **show view**-re.

```
public function show($id): RedirectResponse|View
{
    $dish=Dish::find( id: $id ) ;
    if(!$dish){
        return redirect(to: '/')->with(key: 'fail',value: "no dish found");
    }

    $alg= Allergen::all();
    $course= Course::all();
    return view(view: "show", data: ['alg'=>$alg, 'course'=>$course,"dish"=>$dish]);
}
```

# Backend

- **Az edit function** szinte ugyanaz mint a **show function**, csak itt már nem kell megvizsgálni hogy létezik-e a rekord, mivel a szerkesztés a **show view**-en keresztül érhető el.

- 

```
public function edit($id): View
{
    $dish=Dish::find( id: $id );
    $alg= Allergen::all();
    $course= Course::all();

    return view('edit', data: ['alg'=>$alg, 'course'=>$course, 'dish'=>$dish]);
}
```

# Backend

- Az **update function** hasonló a **show function**-hoz.
- Először is megkeresi a rekordot az id alapján, majd fut egy validálás.

```
public function update(Request $request, $id, ): RedirectResponse
{
    $dish= Dish::find(id: $id);

    $request->validate(rules: [
        'name' =>'required|min:3',
        'courseId'=>'required',
        'desc'=> 'string',
        'img'=> 'nullable|mimes:png,jpg,jpeg',
        'inst'=> 'array',
        'ing'=>'array',
        'allergens'=>'array'
    ]);
}
```

# Backend

- Ezután megnézi, hogy jött-e a requestbe kép, ha igen, akkor törli az előzőt, majd létrehoz egy újat ugyanúgy mint a create-ben.
- Ez a módszer megelőzi a már nem használt képek tárolását

```
if($request->has(key: 'image')){  
    $imgpath= public_path(path: 'dishimg/'.$dish->img);  
    if (File::exists(path: $imgpath)) {  
        File::delete(paths: $imgpath);  
    }  
    $img =$request->file(key: 'image');  
    $ext= $img->getClientOriginalExtension() ;  
    $filename=time().'.'.$ext;  
    $imgpath='dishimg/';  
    $img->move(directory: $imgpath,name: $filename);  
}
```

# Backend

- Majd az update-eli a **dish tábla** többi adatát
- Végül miután törli azokat a rekordokat ahol a dishid megegyezik a \$dish id-jával, ugyanúgy mint a create-ben létrehozta a requestben kapottak alapján.
- Ez a módszer megelőzi a duplikált adatokat, valamint nem marad az adatbázisban rekord hogyha a felhasználó kiveszi onnan.
- 

```
$dish->update([
    'name'=> $request['name'],
    'courseId'=>$request['courseId'],
    'desc'=>$request['desc'],
    'img'=>$filename,
    'inst'=> $request['instructions'],
    'ing'=>$request['ingredients'],
]);
```

```
alg_dish::where(column: 'dishid', operator: $id)->delete();
if(empty($request['allergens'])){
    return redirect(to: 'dish/'.$dish->id)->with(key: 'success',value: '');
}
else{
    foreach ($request['allergens'] as $a) {
        alg_dish::create(attributes: [
            'dishid'=>$dish->id,
            'alg' => $a,
        ]);
    }
}
return redirect(to: "dish/$id");
```

# Backend

- A **destroy function** ugyanúgy mint a szerkesztés, szintén a **show view**-en keresztül érhető el.
- Mivel maga a rekord törlése nem törli a **public/dishimg**-ből a képet, így ha létezik a fájl, az **unlink function**-el elérhető annak is a törlése.
- Végül redirect-el a **front view**-ra.
- kép törlése egy [youtube videó](#) alapján lett elkészítve

```
public function destroy(string $id): RedirectResponse
{
    $dish=Dish::find(id: $id);
    $imgpath= public_path(path: 'dishimg/'.$dish->img);
    if (File::exists(path: $imgpath)) File::delete(paths: $imgpath);

    $dish->delete();

    return redirect(to: '/')->with(key: 'success');
}
```

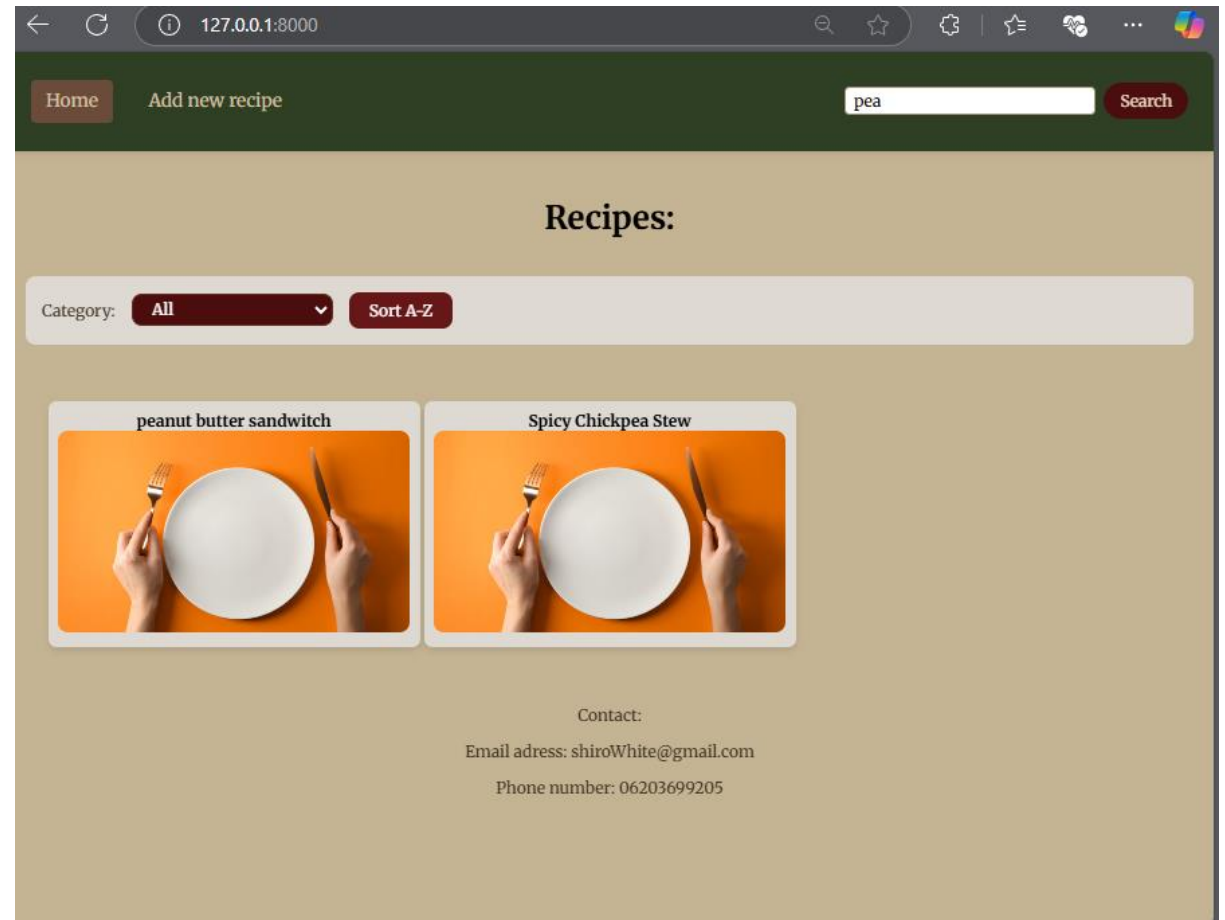
# Frontend

- Laravel blade: miért ezt használtuk?
- **-egyszerű szintaxis:** szimpla és könnyen olvasható a szintaxisa
- **-Dinamikus html generálás:** lehetővé teszi a dinamikus adatkezelést
- **-Újrahasználható kódrészletek:** A Blade sablonok segítségével moduláris és újrahasználható kódot hozhatunk létre, amely javítja a fejlesztési folyamat hatékonyságát .



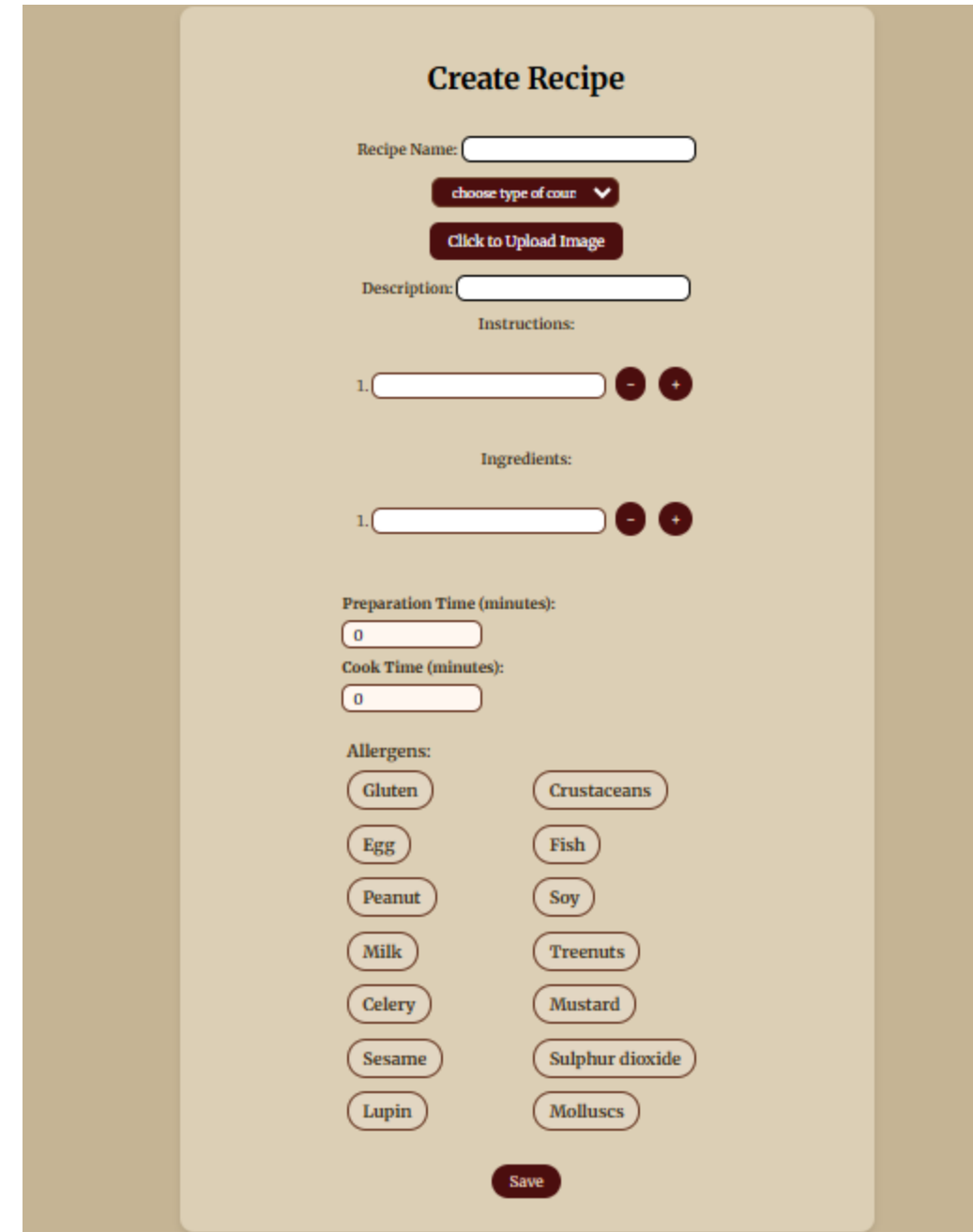
# Kezdőlap

- Itt jelenlátható a weboldal legtöbb funkciója
- A receptek közt lehet filtrálni a recepteket kategória alapján, abc alapján és emellett van keresési funkció.



# Recept feltöltése

- Ebben a szekcióban történik meg a recept feltöltés.
- Sok funkciókkal rendelkezik, részletes feltöltésre van lehetőség.




The image shows a 'Create Recipe' form with a light beige background. The form includes the following fields and controls:

- Create Recipe** (Section Header)
- Recipe Name:** A text input field.
- choose type of cour:** A dropdown menu.
- Click to Upload Image:** A button.
- Description:** A text input field.
- Instructions:** A section header.
- 1.** A text input field for the first instruction, followed by minus and plus buttons for editing.
- Ingredients:** A section header.
- 1.** A text input field for the first ingredient, followed by minus and plus buttons for editing.
- Preparation Time (minutes):** A text input field with the value '0'.
- Cook Time (minutes):** A text input field with the value '0'.
- Allergens:** A section header.
- Allergen Selection:** A grid of buttons for selecting allergens: Gluten, Crustaceans, Egg, Fish, Peanut, Soy, Milk, Treenuts, Celery, Mustard, Sesame, Sulphur dioxide, Lupin, and Molluscs.
- Save:** A button at the bottom right.

# Megjelenítés

- A lap alján látható, hogy ebben a szekcióban lehetőséged van szerkeszteni, vagy a receptet törölni.

## Spicy Chickpea Stew



**Description:** A hearty and flavorful stew packed with protein-rich chickpeas, vibrant vegetables, and a kick of spice. Perfect for a cozy dinner or meal prep.

**Course:** soup

**Instructions:**

1. Stir in diced bell pepper and cook for 2-3 minutes.
2. Add chickpeas, diced tomatoes, vegetable broth, cumin, smoked paprika, chili powder, salt, and pepper.
3. Bring the mixture to a boil, then reduce heat and let it simmer for 15-20 minutes, stirring occasionally.
4. Taste and adjust seasoning as needed.
5. Serve hot, garnished with fresh cilantro if desired.

**Ingredients:**

- 1 tsp ground cumin
- 1 tsp smoked paprika
- 1/2 tsp chili powder
- 2 tbsp olive oil
- Salt and pepper to taste
- Fresh cilantro for garnish (optional)

**Preparation Time:** 10 minutes

**Cook Time:** 20 minutes

**Allergens:**

Egg Soy

[edit recipe](#) [delete recipe](#)

# Megjeleítés

- emellett, ha kép feltöltésére nem kerül sor, akkor egy placeholder kép jelenik meg, ami átveszi a jelenleg nem létező kép helyét, de csak addig míg egy másik kép át nem veszi a helyét.



# Megjelenítés

- Ha a szerkesztés gombra kattintasz, akkor egy borzalmasan hasonló oldalra visz át, amelyben az általad megadott mezőkben megjelennek a receptről megadott információk. Nyilván ilyen esetben ahol elég hosszú a recept, nem fért bele egy képernyőképbe itt lehetőségre kerül az adatok szerkesztése, ami a mentés gomb után a megadott információval átíródik.
- A mentés után pedig vissza kerülünk a receptünkhöz.

3.  -

4.  -

5.  -

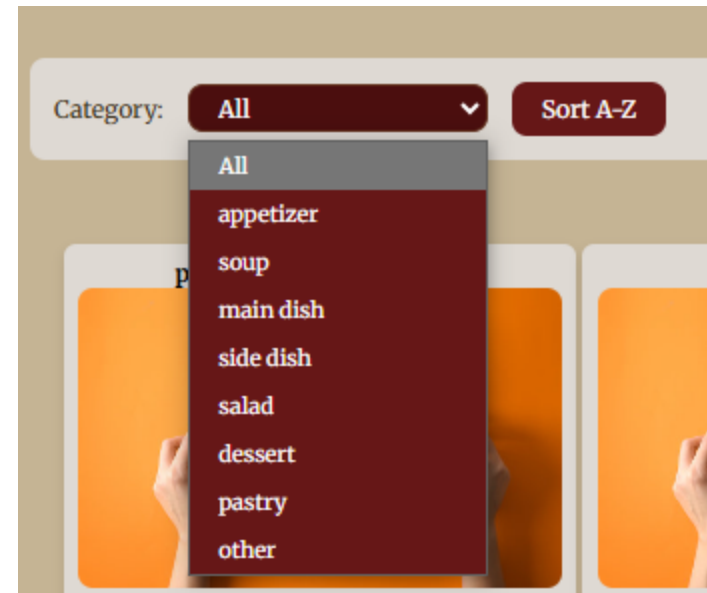
6.  - +

Allergens:

<input type="checkbox"/> Gluten	<input type="checkbox"/> Crustaceans
<input checked="" type="checkbox"/> Egg	<input type="checkbox"/> Fish
<input type="checkbox"/> Peanut	<input checked="" type="checkbox"/> Soy
<input type="checkbox"/> Milk	<input type="checkbox"/> Treenuts
<input type="checkbox"/> Celery	<input type="checkbox"/> Mustard
<input type="checkbox"/> Sesame	<input type="checkbox"/> Sulphur dioxide
<input type="checkbox"/> Lupin	<input type="checkbox"/> Molluscs

# Megjelenítés

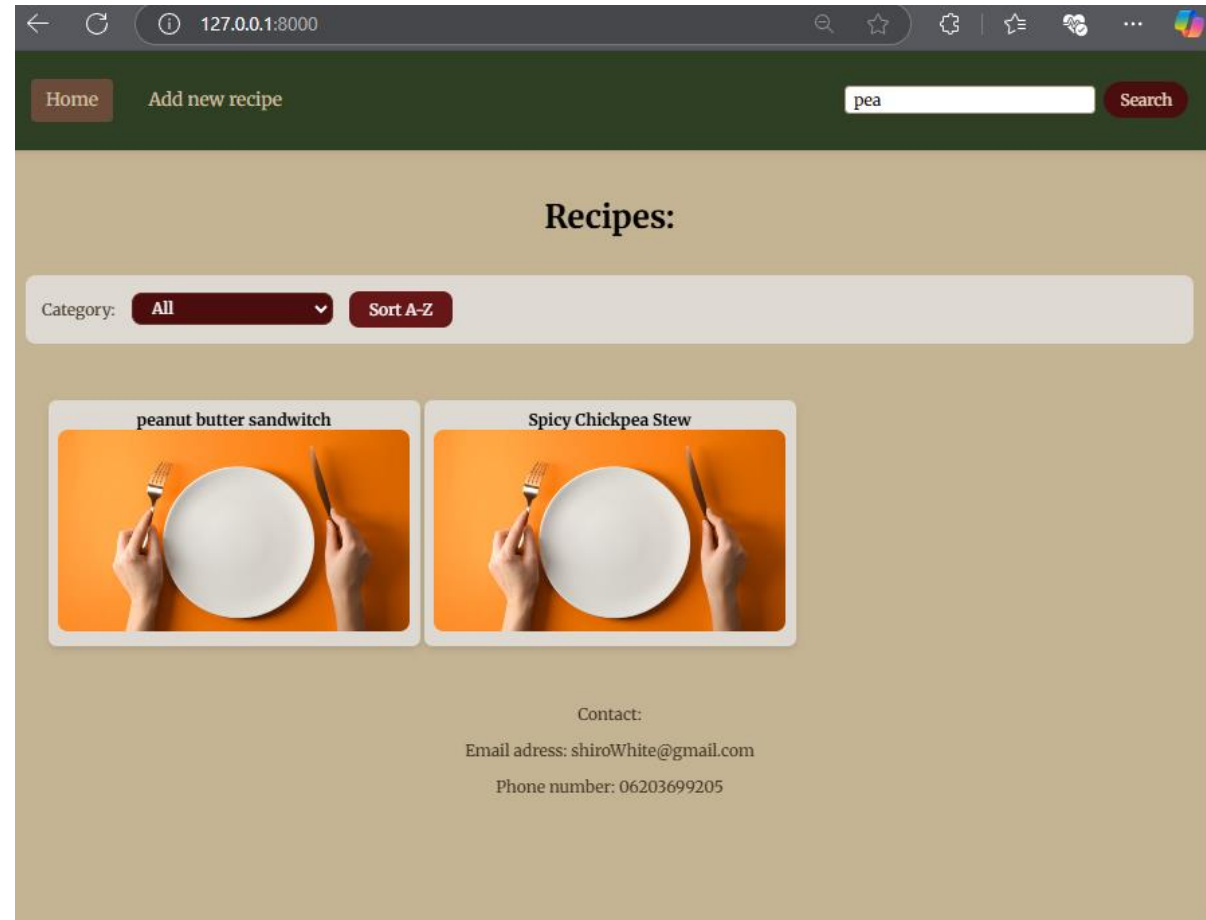
- Visszatérve a főoldalunkhoz, ha szeretnénk mást csinálni, mint recepteket feltölteni, akkor van esélyünk a receptek böngészéséhez egy maréknyi segítség, mellyel könnyebbé tehetjük a dolgunkat.
- ebbe segíthet a filterezési kategóriánk:
- Ebben a kategóriába lehetőségünk van arra, hogy vagy A-tól Z-ig rendezzük a recepteket, vagy rákattintva Z-től A-ig rendezze a recepteket.
- Emellett ugye szortírozni lehet kategóriák szerint, ugyan azok a kategóriákból amelyet láthattál a recept feltöltéseknél:
- 



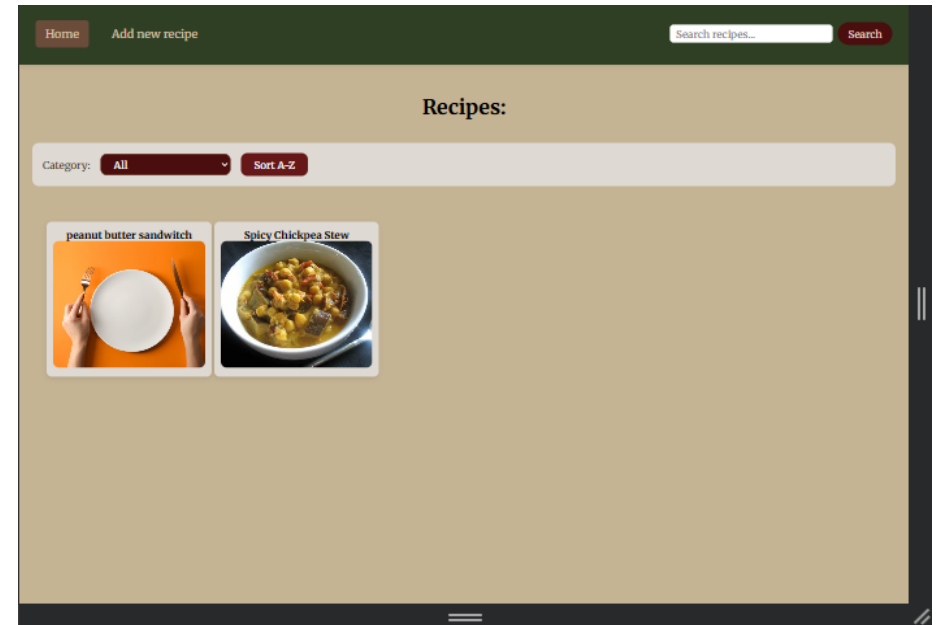
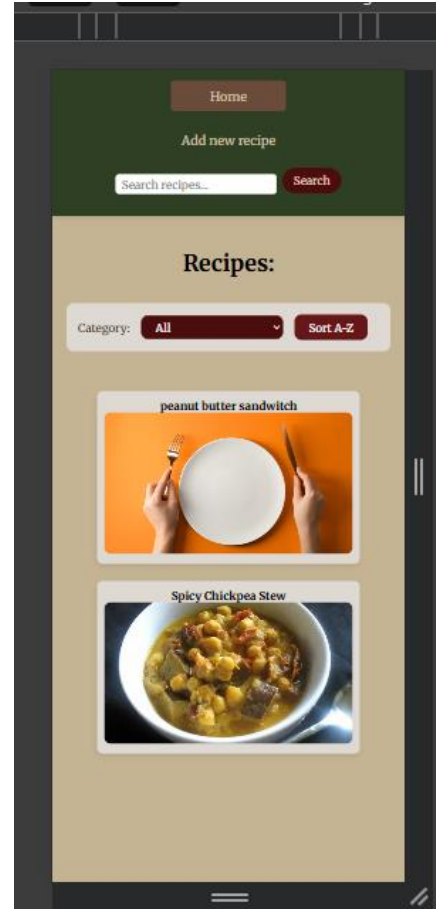
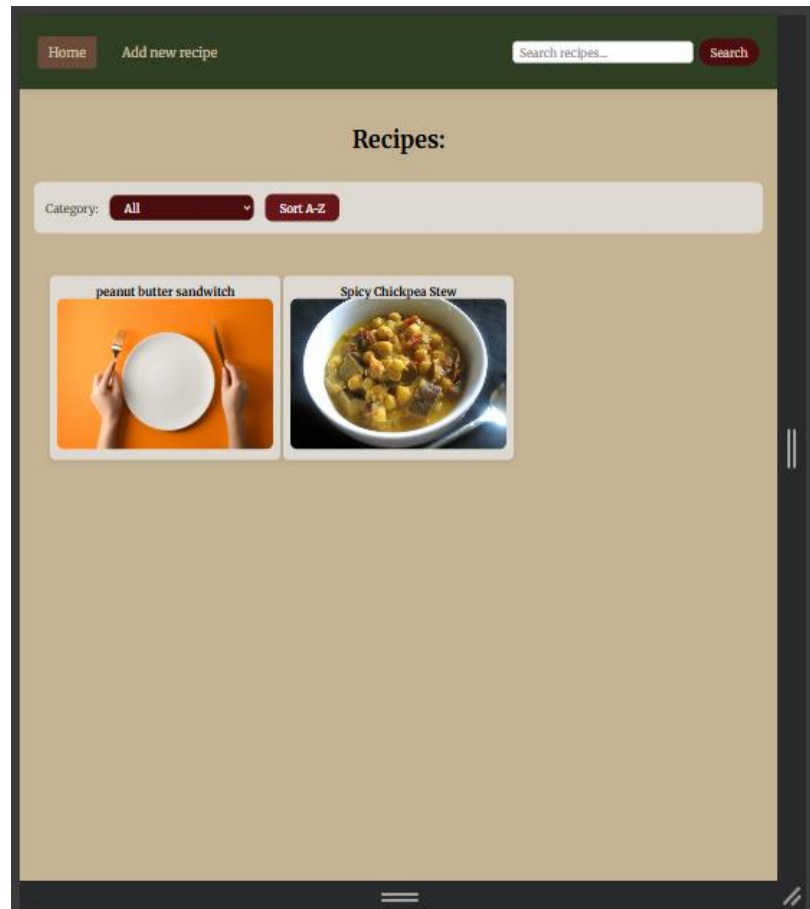
# Megjelenítés

- És ha ezek a funkciók nem lennének kielégítőek, akkor hasznodra veheted a keresési funkciót, amit a fejlécbe találsz.
- beírt szöveg után, a “keresés” megnyomására kiadja neked a legmegfelelőbb találatokat. Emellett ha nem úgy érzed, hogy meg akarod nyomni a “keresés” gombot, akkor szimplán nyomhatsz egy entert ami ugyanezt az eredményt fogja teljesíteni.

- 



# Reszponzivitás





# Csapatmunka

- Github
- Discord

# Csapatmunka

- Kovács Orsolya – backend
  - Adatbázis kezelés és szerkesztés
  - recept törlése, szerkesztése
- Lele Regina – frontend
  - Receptek megjelenítése
  - Adatok feltöltése
  - Filterek és kategóriák

# Publikáció

- Nyilvános link: [kovkrist5/recipe-website](https://github.com/kovkrist5/recipe-website)
- Minden megtalálható githubon.

Köszönjük a figyelmet!