Compte rendu itération 3

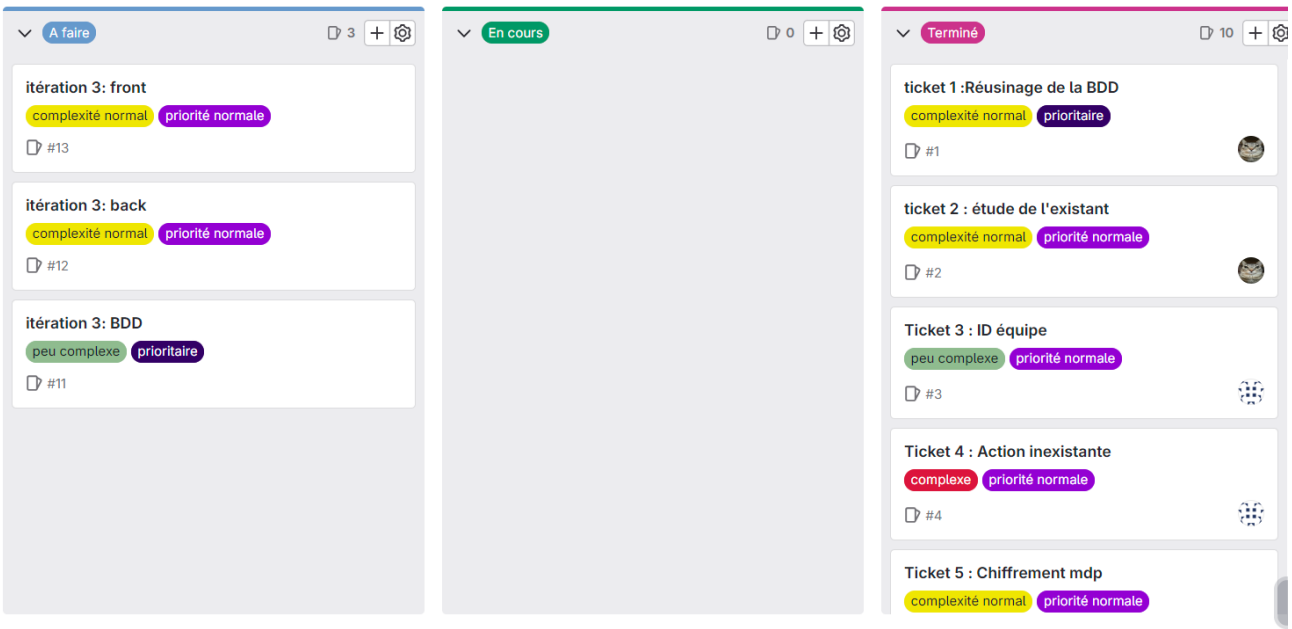
Worms Loric – Rialland Stefen – Thomas Leroux

Equipe F - https://gitlab.com/LoricWorms/p1_gf_siteresto2024

Table des matières

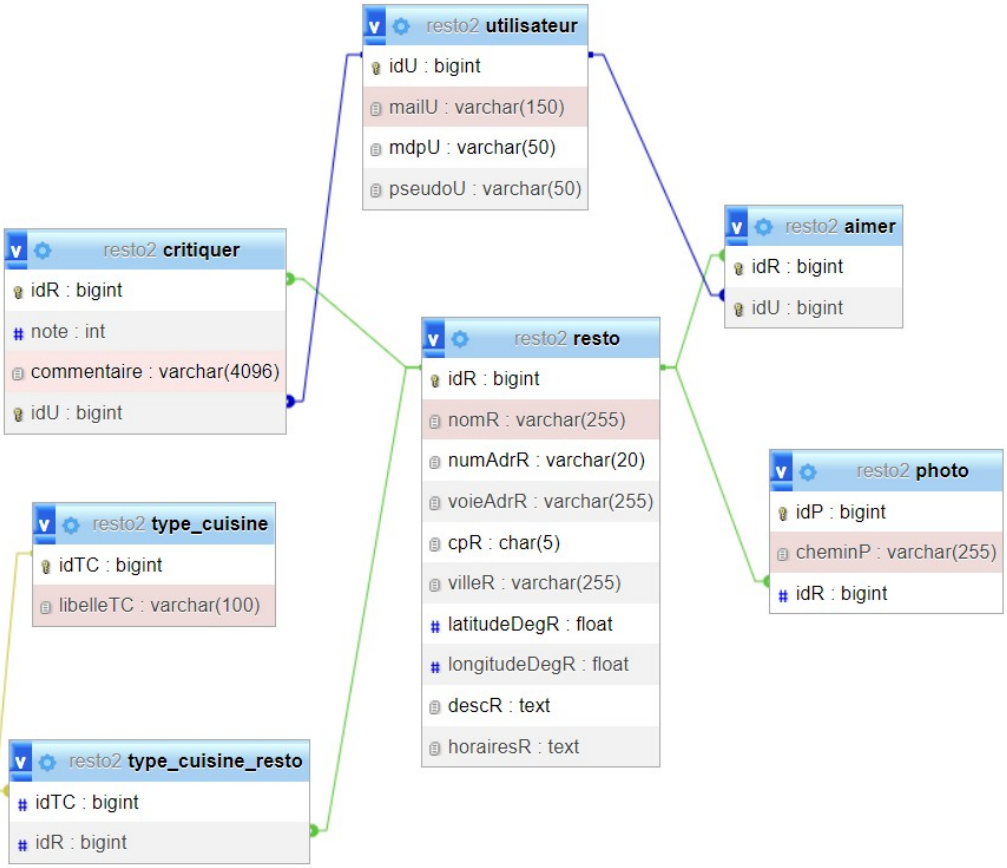
compte-rendu_réunion (28/10/2024)

Tout les tickets ont été finis lors de la séance précédant sans difficulté notable, nous avons même eu le temps de commencer l'itération 3. Nous avons repérés 3 tâches distinctes : la modification de la BDD (Stefen), celles du back (Loric) et du front (Thomas).



compte-rendu technique

I- Ajout à la BDD



2 nouvelles tables : « type_cuisine » répertoriant les différents type de cuisine et « type_cuisine_resto » qui permet de faire le lien entre les restaurants et le type de cuisine qu'ils proposent.

II- La classe métier

```php
namespace modele\metier;

/**
 * Description of TypeCuisine
 *
 * @author loric
 */
class TypeCuisine {
    //put your code here
    private  int $idTC;
    private  int $idR;
    private string $libelle;

    public function __construct(int $idTC, int $idR, string $libelle) {
        $this->idTC = $idTC;
        $this->idR = $idR;
        $this->libelle = $libelle;
    }
```

Création d'une nouvelle classe métier « TypeCuisine » composé d'un constructeur pour créer des objets « TypeCuisine » ainsi que des accesseurs et mutateur pour interagir avec les données.

III- La classe DAO

```php
class TypeCuisineDAO {
    //put your code here
        /**
     * Retourne un une liste d'objet TypeCuisine
     * @return lesTypeCuisine la liste d'objet TypeCuisine recherché ou null
     * @throws Exception transmission des erreurs PDO éventuelles
     */
    public static function getAllTypeCuisineResto(): array {
        $lesTypeCuisine = [];
        try {
            $requete = "SELECT * FROM type_cuisine_resto INNER JOIN type_cuisine ON type_cuisine.idTC = type_cuisine_resto.idTC";
            $stmt = Bdd::getConnexion()->prepare($requete);
            $stmt->execute();

            // Vérifiez si un enregistrement a été trouvé
            if ($stmt->rowCount() > 0) {
                // Pour chaque enregistrement
                while ($enreg = $stmt->fetch(PDO::FETCH_ASSOC)) {
                    //Instancier un nouveau type de cuisine et l'ajouter à la liste
                $lesTypeCuisine[] = new TypeCuisine($enreg['idTC'], $enreg['idR'], $enreg['libelleTC']);
                }
            }
        } catch (PDOException $e) {
            throw new Exception("Erreur dans la méthode " . get_called_class() . "::getTypeCuisineById : <br/>" . $e->getMessage()
        }
        return $lesTypeCuisine;
    }
}
```

Création d'une nouvelle classe DAO « TypeCuisineDAO » permettant de récupérer les données de la BDD et de créer un objet « TypeCuisine » pour chaque enregistrement de la table « type_cuisine_resto ».

VI- Modification des contrôleurs

```php
use modele\dao\TypeCuisineDAO;

$listeTypeCuisine = TypeCuisineDAO::getAllTypeCuisineResto();
```

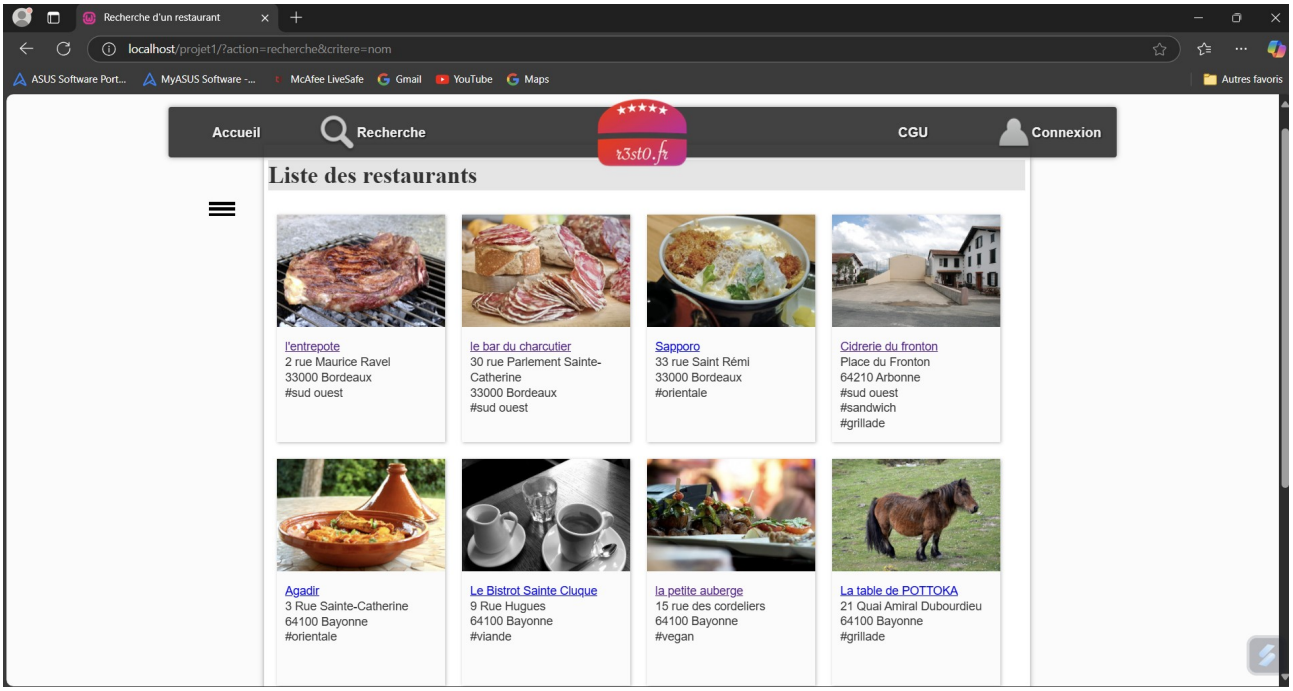
Ajout d'une variable contenant la liste des objets « TypeCuisine » récupérer via la méthode « getALLTypeCuisineResto() » dans les contrôleurs « accueil.php », « rechercheResto.php » et « detailresto.php ».

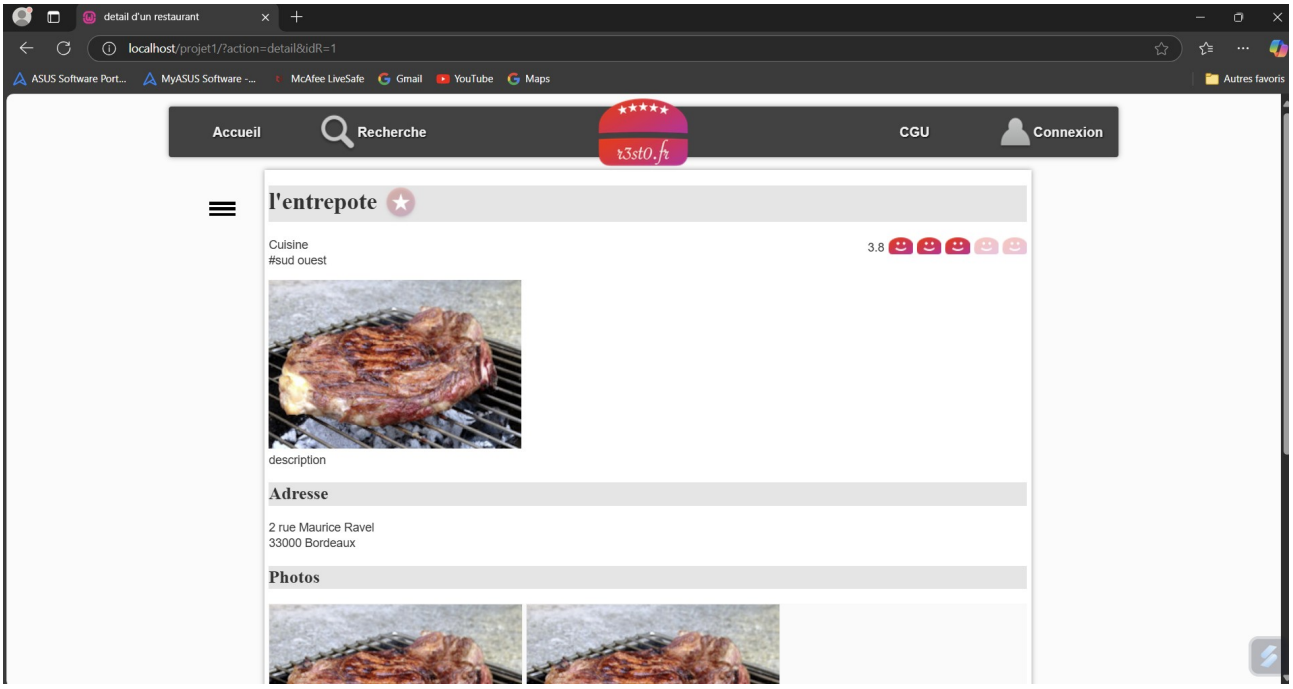
V- Modification des vues

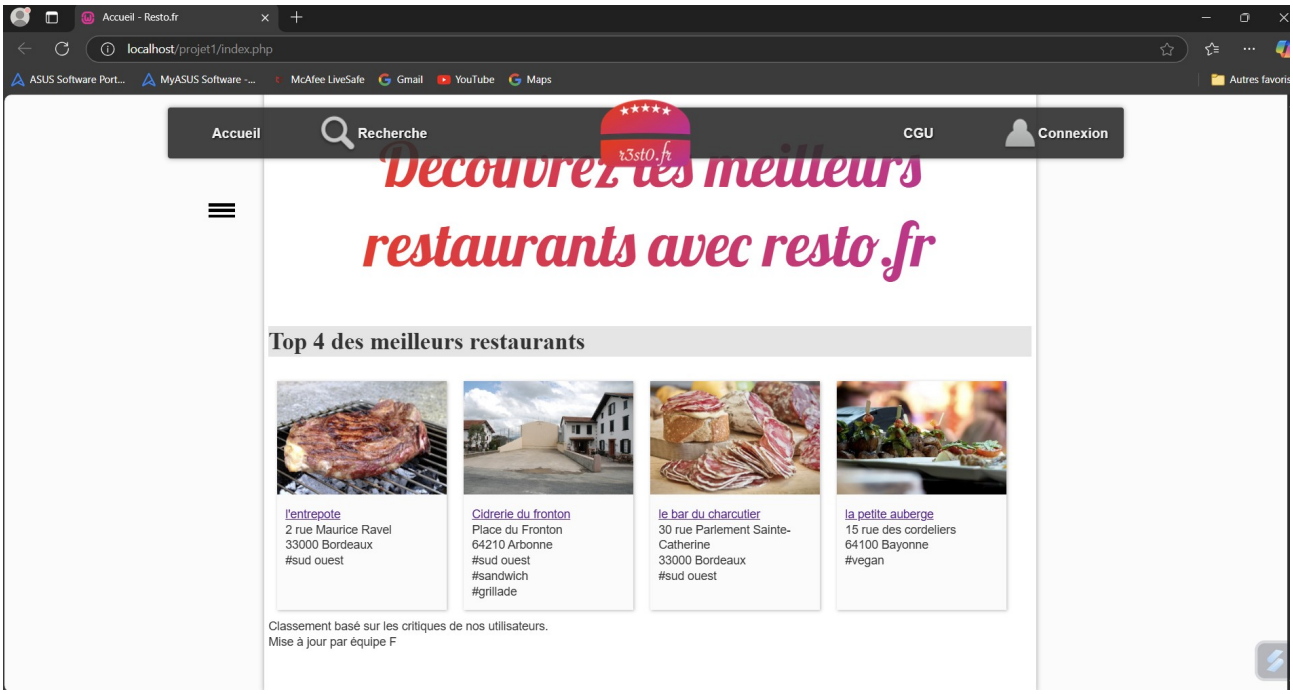
```php
<?php

foreach ($listeTypeCuisine as $unTypeCuisine) {
    if ($unTypeCuisine->getIdR() == $unResto->getIdR() ) { ?>
        <br/>
         #<?= $unTypeCuisine->getLibelle() ?>

    <?php
    }
}

?>
```

Ajout d’une boucle qui parcourt « $listeTypeCuisine » et qui affiche le libelle du type de cuisine si il y a concordance entre l’id de « $unResto » et l’id de « $unTypeCuisine ».

VI- Résultat







note : cependant les critiques n’apparaissent plus sur la page du détail des restaurants.