Bonjour je m’appel Jean Baptiste Servais j'ai 24 ans et je vais vous présenter la plateforme nutella.

Nous allons vous présentez ce projet selon la logique d'un utilisateur qui découvre le site pour la première fois. L'utilisateur va faire une première recherche sans s'etre inscrit. Nous lui proposeront alors de s'inscrire. De ce fait, nous vous présenteront l'inscription d'un utilisateur à travers les différents mécanismes, codes mise en place. Ainsi l'utilisateur peut se connecter. Nous vous présenteront alors la connexion. Une fois connecté, l'utilisateur peut enfin enregistrer des aliments. Une nouvelle fonctionnalité apparaît qui est la fonction de remplacer un aliment.

Pour introduire notre application il faut que vous sachiez que toutes les pages html incluent une navbarre (navebarre.html) et un bottom(bottom.html) grâce au gabarit d'inclusion. Elles sont toutes reliées ou link à un fichier js et css.

Pour commencer, nous allons vous présenter une première recherche d'un utilisateur qui n'est pas inscrit. Voici la première recherche de l'utilisateur (slide de simulation web). Par exemple ici il cherche l'aliment céréactive. Nous lui présentons alors l'aliment recherché et 5 autres aliments de nutriscore les plus élevés de la catégorie. Nous lui indiquons qu'il faut qu'il se connecte qui sous-entend qui doit s'inscrire afin de pouvoir enregistrer des aliments.

Comment avons nous fait cela ? L'utilisateur tape « céréactive » dans l'input de la navbarre. Ici. Ou a partir de la page home.html (1b – la recherche). Nous utilisons une balise input englobée d'un form qui va envoyer une requête vers la fonction search() (1c- la recherche).

Ici, si on récupère une requete POST soit la recherche de l'utilisateur céréactive. On cherche search, username ou validate. Ici on va se concentrer sur la variable search.

Si on recoit une requete POST de la balise input ayant le nom cool soit de la navbarre soit de la page home alors on utilise ce bloc d'instruction (1c la recherche).

On récupère l'utilisateur connecté, on va appeler controle_data_food(), qui va nous renvoyer un True ou False. Si la variable stock nous renvoie False alors on redéfinis la variable exceeded_stock qui initialement est vide soit deux guillemets.
Dans le cas présent notre utilisateur n'est pas connecté alors nous n'appelons pas les instructions if .

Nous appelons également les fonctions image_food() et title_food() que nous stockons dans des variables. Afin de justifier ce try except : ce sont les test nous ont conseillé. Ensuite nous appelons better_nutri() qui permet de pouvoir afficher les 5 autres aliments de meilleur nutriscore. Ces fonctions se servent du parametre de recherche soit la variable search c'est a dire céréactive qui a été introduit dans les inputs html par l'utilisateur. Afin de pouvoir afficher les informations nous les retournons sur le template recherche.html.

La première fonction est Controle_data_food() qui fait appel à la table foodaccount (que nous verront plus tard). On prend l'username de l'user en connexion en parametre. Nous récupérons tous ces aliments. S'il a plus de 6 aliments nous retournons nombre de produit sup a 6 sinon stockage produit possible.

Ensuite nous appelons la fonction image_food() qui va pouvoir afficher l'image de la recherche. Nous appelons dans la table aliment ou nous faisons une requete a l'orm django son équivalent en sql est « select * from aliment where name_aliment $LIKE$ » . Nous lui demandons l'image de l'aliment du paramètre para et le retournons.

Nous cherchons maintenant le titre de l'aliment de la recherche avec la même opération. Pourquoi avoir séparé en 2 fonction ? Pour une meilleur lisibilité au détriment du poids de l'application.

Nous appellons maintenant la fonction better_nutri(). Nous demandons a l'orm django de recuillir les informations de l'aliment correspondant au paramètre para. Nous sélectionnons dans une liste le nom de l'aliment, son id_categorie, son nutriscore son image et son id. Dans la table category nous demandons par une boucle for les 20 premieres aliments leur nom, idcategory, nutriscore et image par ordre croissant de nutriscore.

Voila comment on fait une recherche d'aliment. On récupère l'utilisateur en cours. On va chercher dans la table foodaccount les lignes qui correspondent à la ligne username pour savoir le nombre de produit (cela va pouvoir permettre d'introduire plus tard search() en étant connecté) mais aussi l'image, le titre de la recherche et enfin pouvoir afficher 5 autres aliments ayant le meilleur nutriscore de la catégorie.

Maintenant voici la page ou l'utilisateur s'inscrit. Il rentre les informations suivantes : speudo, mail1 mail2 et son mot de passe. Il doit appuyer sur valider.

Afin de pouvoir faire cela nous avons appelé un formulaire. Le formulaire de base de django Model_user et appelons les champs suivant : username, email, email2 et password. Nous les nettyons et creons des erreur (les meme emial ou si l'email existe deja).

Ensuite nous retournons dans views.py. Si le formulaire est valide (sans erruer) alors on nettoie le mot de passe, le redéfinissons et sauvegardons l'utilisateur. Nous enregistrons l'username de l'user et lui définissons 6 aliments dans la table foodAccount. Nous le redirigeons vers home.html.

Voici foodAccount. Un name, et ces 6 aliments.

Maintenant l'utilisateur se déconnecte en appuyant sur le dernier logo a gauche en appelant la fonctio logout_view

Les formulaires se présentes sous cette forme : username et password en charfield. On les nettoie et on les authentifie grace a authenticate de django pour le connecter et le redirigeons vers home.html.

L'utilisateur se connecte. Voici la page. Ici deux champs a remplir. Puis il faut valider. On fait appel a login_view(). Ou nous recupérons une requete POST et si le formulaire et valide (a matche en database) alors on nettoie le mdp et l'username. On utilise authenticate de django pour le connecter et le redirigeons vers home.html.

Si l'utilisateur n'a pas enregistrer daliments et veut voir ces aliments on lu iafficher cette page d'erreur avec un message personalisé.

Donc l'utilisateur fait sa deuxieme recherche avec le but d'enregistrer des aliments. Par exemple ferrerro rocher. Nous lui affichons la recherche et 5 autre aliments.

Voici un des blocks qui servent a sauvegarder l'aliment. Une fonction pushlist onclick() de la checkbox.

On effectue une requete de type ajax ou nous envoyons a la fonction search de views.py l'aliment et l'user en cours.

L'utilisateur appuie on lui affiche une croix verte grace a :

pushlist(). S'il n'y a pas 'vous avez trop..' alors on afficher une croix verte sinon on afficher une croix rouge.

Comment mettons nous vous avez trop.. ? Tout dabord on verifie si on recoit par une requete post ensuite on verifie contrôle data_food ensuite on verifie username et validate. Si oui cad not none alors on verifie que le produit n'est pas deja present puis si verification_produt not_two contenu dans mla variable veri.

Si veri renvoie true alors on insere le produit sinon on affiche un message comme quoi l'utilisateur a deja trop de produit et affichons la croix rouge

Nous avons deja vu controle_data_food(). On verifie que l'utilisateur n'a pas trop daliment.

Verification product not two permet de savoir si l'aliment n'est pas deja present. On appel foodaccount en prenant la table ou le nom de l'utilisateur match. Puis on recupere les aliments et testons chaque aliment. Si le produit est deja present on retourn false sinon True.

Maintenant comme c'est sa premiere mise en aliment on va inserer les aliments avec insert_food.

On appel la table foodaccount ou le nom de l'utilisateur match et verifions chaque table name_aliment. Si la colonne est vide alors on enresitre la selection de l'utilisateur via la checkbox. Sinon on continue jusqu'à une collonne vide.

Maintenant l'utilisateur a 6 aliments. S'il refait une recherche on affiche le message suicant « oups vous avez trop daliment » ce qui permet de mettre la croix rouge et empecher la sauvegarde.

Maintenant l'utilisateur veut voir ses aliments.

Pour cela on recupere l'utilisateur en cours. On utilise my_food_user, et display_food et le retounons dans la page.

My_food_user() ici on recupere la table foodaccount et les lignes correspondant a l'utilisateur. On recupere les name_aliment. On stock tous ca dans une liste et la retournons. Nous la retournons car on appel display_food qui prend en parametre cette liste.

On appel la table aliment. Par une boucle for on recupere dans la table category le nom de laliment, le code produit la description le nutriscore, l'image, le nom du magasin et sa marque. On ajoute ca a une liste puis la retournons.

Maintenant l'utilisateur veut voir la description d'un aliment. Il appuie sur l'image d'un des aliments. On lui affiche 3 choix via la fonction choice() js. Soit le remplacementde produit soit la description ou non rien on reaffiche l'image. Par un effacage de balise et la réecriture de cette balise.

Nan rien on display un block qui etait invisible, on efface les 3 choix pour reafficher l'image.

Pour la description l'utilisateur appuie sur detail aliment puis appuie sur voir la fiche.

Pour cela on appel la fonction food_det()

par l'appuie donc par un submit d'un input on recupere une requete POST. On recupere la data name d'une balise produit appel la fonction food_detail, son url, son nom, son code, son image et constituons l'url de l'image nutriscore et retournons tout cela.

La fonction food_details() appel la table aliment et la ligne correspondant au parametre value.

Maintenant l'utilisateur veut remplacer un aliment. Il appuie sur le dernier chois remplcaer ce produit.

Si l'utilisateur veut rempalcer un produitr par un appuie sur un input alors On recupere un post d'une requete. On recupere la donnée remplace_food. Puis l'utilisateur en cours, verifions la verifcation_replacement(), puis remplacons le produit sinon l'utilisateur a deja le produit et on le renvoie a la page mes_aliments.

Sinon l'utilisateur vient dappuyer sur rempalcer laliment alors on renvoit 6 aliments par image (pour limage du haut, titre pour l'image d'en haut et replace équivalent a better_nutri et renvoyons tout ca pour l'affichage.

Donc la methode verification_replacement() permet de voir si l'utilisateur a deja l'aliment. En appelant la table foodaccount et les lisgne du parametre username. Si un des produits match avec le parametre product on renvoie False sinon True. Si on renvoie true c que laliment n'y est pas.

Alors on utilise data-replace(). On appel foodaccount, et les ligne correspondant au parametre username. Puis on verifie chaque aliment poru voir si ca match avec le parametre product si oui alors on le remplace par new_product.

Construction de la database :

Tout dabors on fait un requete soup a l'url des categorie deopenfactfood. On prend les 3 premeires

categorie. Et les insert par une requete sql.

Ensuite on recupere chaque element des categorie et inserons dans les categorie. Le nom, image ect..

Puis on supprime les virgule, les charactere spécuaux et les data vide.
Rictection les test juste a lire.