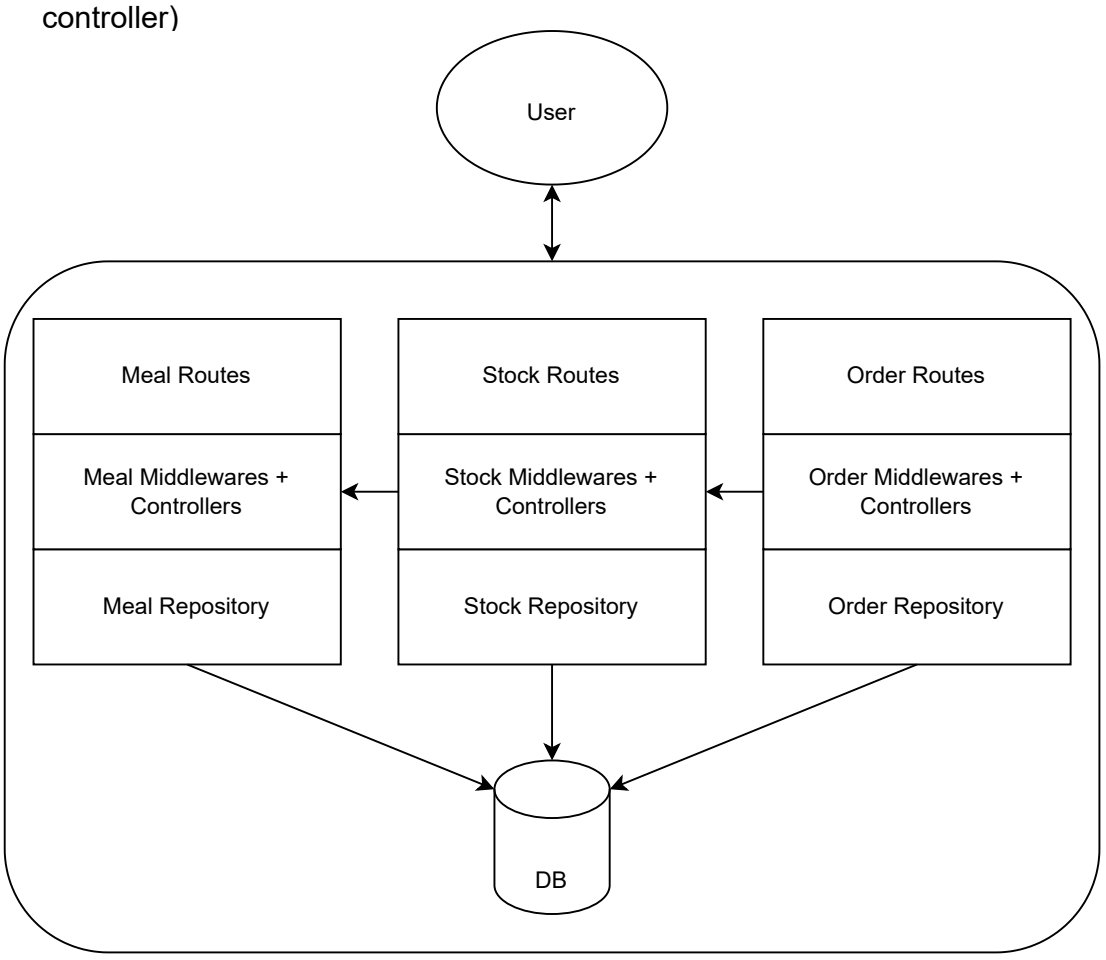
Architecture Initiale

Monolithe

L'architecture initiale serait donc un monolithe découpé en features / services (Stock, meal list, orders, etc...). Nous aurons donc pour chaque feature un fichier route, controller, model, middleware et repository correspondant. L'idée étant qu'il y est le moins de dépendances possible entre les features, et lorsque nécessaire faire en sorte que ça soit le middleware qui dépende du répository d'une autre feature. De ce fait il sera facilement par un appel API si ce répository venait à être externaliser.

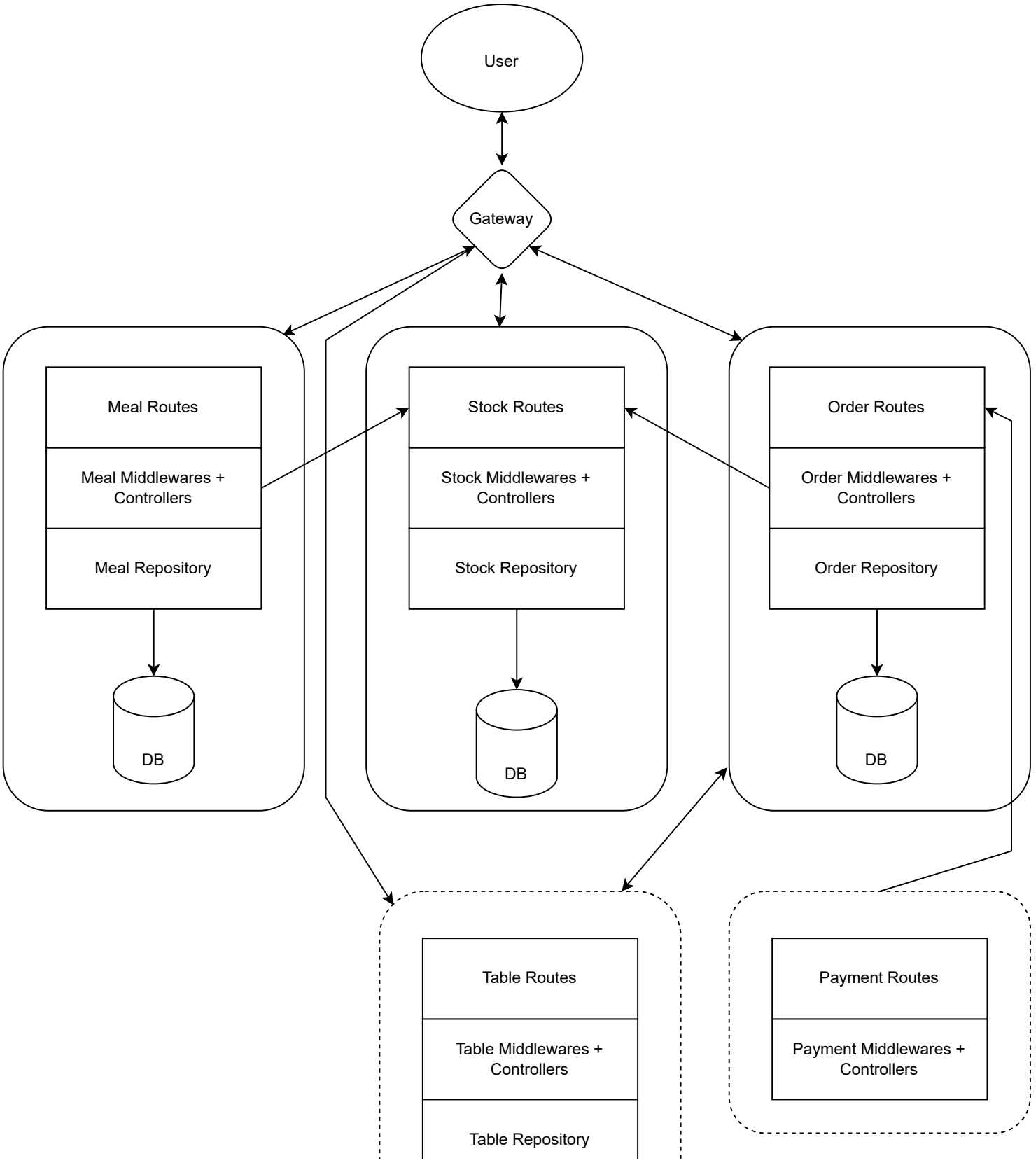
Pour ce qui est de la base de données, il faut s'assurer que chaque table ne sois accessible que par le repository associé et si une dépendance est nécessaire celà doit se faire plus haut (Au niveau du middleware ou du controller)

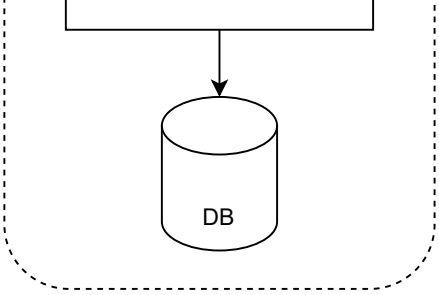


Architecture Cible

Microservices

Lorsque plusieurs équipes différentes seront constitués pour poursuivre le projet, l'architecture microservices serait plus adapté. La mise en place des microservices serait faciliter par le choix pris en amont de structuer le monolithe en features / services et de réduire au maximum les dépendances. Les middlewares dépendant auparavant de repository d'autre feature n'auront qu'a faire un call API au services distant correspondant. Ainsi une nouvelle équipe peut facilement venir ajouter son service dans l'architecture




DB