ok je t'explique mon idée. je veux une application de gestion de dépenses financières. dans la page d'accueil, je veux un calendrier avec afficher le mois actuel. et chaque fois que le user clique sur la date du jour, il y aura une page avec un tableau en formulaire qui sera afficher sur le navigateur et le user pourra entrer ses données. une fois que le user clique sur valider, la page ne se recharge pas mais affiche les données enregistrées en direct. en haut ou en bas de ce tableau, il y aura un bouton retour qui retournera sur la page avec le calendrier. dans cette page d'accueil, je veux aussi en bas, 2 diagrammes: un avec l'historique des dépenses du jour et l'autre avec l'historique des dépenses du mois. dans les historiques il y aura des requêtes pour récupérer les totaux (des jours ou des mois selon le diagramme). je résume ce que je vois sur la page d'accueil est le calendrier et les 2 historiques. je dois aussi te dire que je vais créer 2 entity: User et Depense. dis moi comment je peux créer ma page d'accueil et dis moi si c'est mieux de faire des controllers ou des actions? reponse en tiret et breve

ChatGPT

1. **Page d'accueil Symfony**:
   * Crée un contrôleur principal pour la page d'accueil (par exemple, AccueilController).
   * Ajoute une action dans ce contrôleur pour gérer la page d'accueil.
2. **Calendrier**:
   * Utilise un package JavaScript pour afficher le calendrier (par exemple, FullCalendar.js).
   * Crée une route et une action pour gérer les interactions avec le calendrier.
3. **Tableau en formulaire**:
   * Crée une route et une action pour afficher le formulaire de saisie des données.
   * Gère la validation du formulaire et l'enregistrement des données.
4. **Diagrammes**:
   * Utilise une bibliothèque JavaScript pour les diagrammes (par exemple, Chart.js).
   * Crée une action pour récupérer les données nécessaires (totaux des dépenses) via des requêtes et les transmettre à la vue.
5. **Retour au Calendrier**:
   * Ajoute un bouton de retour dans la vue du formulaire qui redirige vers la page avec le calendrier.
6. **Structure des Contrôleurs/Actions**:
   * Crée des actions distinctes pour chaque fonctionnalité (calendrier, formulaire, historiques).
   * Organise les actions dans le contrôleur principal (AccueilController) pour maintenir une structure claire.

L'approche des contrôleurs et des actions est recommandée, avec des actions distinctes pour chaque fonctionnalité afin de garder le code organisé et modulaire.