Le projet nécessite la mise en place d'un système robuste de gestion de comptes avec des rôles distincts tels que "User", "Admin", "Producteur" et "CommManager", chacun ayant des droits spécifiques. Cette exigence suggère l'utilisation d'une solution d'authentification et d'autorisation solide, notamment ASP.NET Identity pour ASP.NET Core, qui offre une gestion flexible des rôles et des droits.

Pour la persistance des données, l'utilisation d'une base de données relationnelle (BDD) semble appropriée. Microsoft SQL Server, MySQL ou PostgreSQL pourraient être des choix pertinents en fonction des préférences et des contraintes spécifiques du projet. Les entités principales à stocker pourraient inclure des données relatives aux jeux ("Games"), aux favoris ("Favorites"), aux comptes utilisateurs avec leurs droits associés ("Account"), et aux actualités ("News").

En termes de modélisation de données, il serait judicieux de définir des relations appropriées entre les différentes entités. Par exemple, un utilisateur peut avoir plusieurs jeux favoris, et un jeu peut être associé à plusieurs comptes utilisateurs.

Du point de vue du développement front-end, l'utilisation d'un framework JavaScript moderne comme React ou Angular pourrait améliorer l'expérience utilisateur en permettant des mises à jour en temps réel et une interaction fluide avec le backend.

La communication entre le frontend et le backend pourrait se faire via des API RESTful, ce qui simplifierait l'intégration de diverses fonctionnalités tout en assurant une séparation claire entre les couches front-end et back-end.

En résumé, pour répondre aux besoins spécifiés, la mise en place d'une architecture ASP.NET Core côté serveur, associée à une base de données relationnelle, des services d'authentification et d'autorisation avancés, ainsi qu'une interface utilisateur réactive, pourrait constituer une solution technique solide pour ce projet.