Strukturyzowany i uporządkowany kod: Klasy takie jak HomeView, ArticleDetailView, AddPostView, organizują kod wokół celu widoku, co czyni go bardziej czytelnym i łatwiejszym w utrzymaniu.

Ponowne wykorzystanie i rozszerzanie: Dziedzicząc z ogólnych widoków Django (np. ListView, DetailView, CreateView), wykorzystujesz gotowy kod, zmniejszając redundancję.

Wyraźne oddzielenie odpowiedzialności: Każda klasa widoku jest odpowiedzialna za konkretny aspekt aplikacji (lista postów, dodawanie komentarzy, tworzenie kategorii), co jest zgodne z zasadą separacji obowiązków.

Dostosowanie i elastyczność: CBVs pozwalają na łatwe dostosowanie. Na przykład, w AddPostView i AddCommentView, możesz zmodyfikować metodę form\_valid, aby dodać niestandardowe zachowanie, co pokazuje elastyczność CBVs.

Integracja z funkcjami Django: Twoje wykorzystanie reverse\_lazy dla URL-i sukcesu oraz formularzy (takich jak PostForm, EditForm) pokazuje, jak CBVs bezproblemowo integrują się z innymi funkcjami Django, zwiększając funkcjonalność i doświadczenie użytkownika.

Te punkty podkreślają korzyści płynące z użycia Widoków Klasy w Django, czyniąc aplikację internetową bardziej efektywną, uporządkowaną i skalowalną.