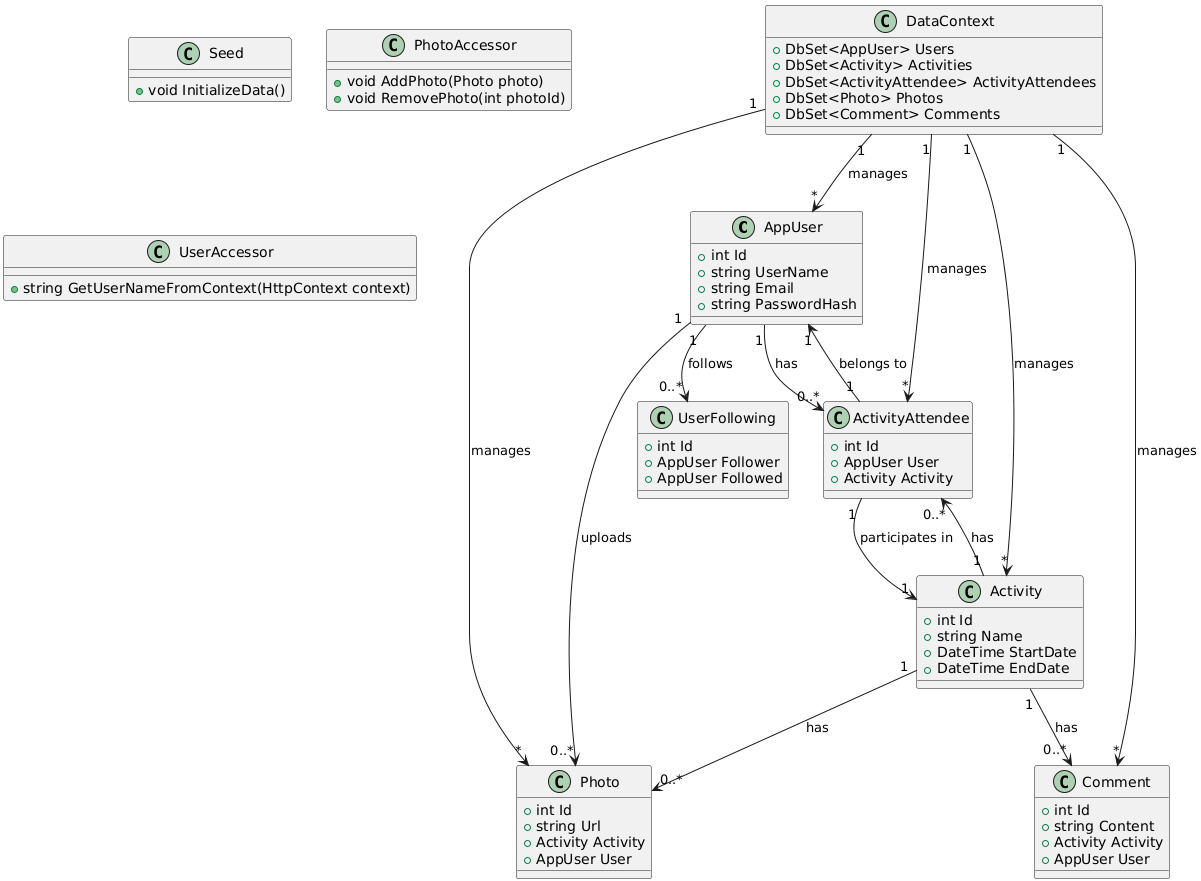
https://www.plantuml.com/plantuml/uml/

## Diagram klas (Class Diagram)



Kod:

@startuml

class AppUser {

+int Id

+string UserName

+string Email

+string PasswordHash

}

class Activity {

+int Id

+string Name

+DateTime StartDate

+DateTime EndDate

}

class ActivityAttendee {

+int Id

+AppUser User

+Activity Activity

}

class Photo {

+int Id

+string Url

+Activity Activity

+AppUser User

}

class Comment {

+int Id

+string Content

+Activity Activity

+AppUser User

}

class UserFollowing {

+int Id

+AppUser Follower

+AppUser Followed

}

class DataContext {

+DbSet<AppUser> Users

+DbSet<Activity> Activities

+DbSet<ActivityAttendee> ActivityAttendees

+DbSet<Photo> Photos

+DbSet<Comment> Comments

}

class Seed {

+void InitializeData()

}

class PhotoAccessor {

+void AddPhoto(Photo photo)

+void RemovePhoto(int photoId)

}

class UserAccessor {

+string GetUserNameFromContext(HttpContext context)

}

AppUser "1" --> "0..\*" ActivityAttendee : has

Activity "1" --> "0..\*" ActivityAttendee : has

ActivityAttendee "1" --> "1" AppUser : belongs to

ActivityAttendee "1" --> "1" Activity : participates in

Activity "1" --> "0..\*" Comment : has

AppUser "1" --> "0..\*" UserFollowing : follows

Activity "1" --> "0..\*" Photo : has

AppUser "1" --> "0..\*" Photo : uploads

DataContext "1" --> "\*" AppUser : manages

DataContext "1" --> "\*" Activity : manages

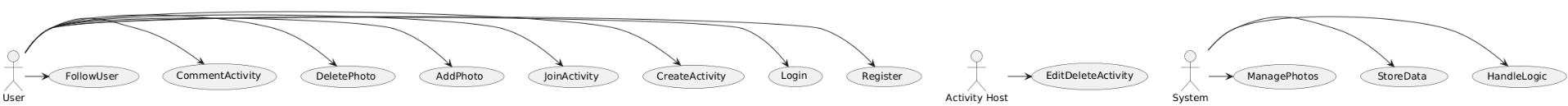
DataContext "1" --> "\*" ActivityAttendee : manages

DataContext "1" --> "\*" Photo : manages

DataContext "1" --> "\*" Comment : manages

@enduml

## Diagram przypadków użycia (Use Case Diagram)



Kod:

@startuml

actor User

actor Host as "Activity Host"

actor System

User -> (Register)

User -> (Login)

User -> (CreateActivity)

User -> (JoinActivity)

User -> (AddPhoto)

User -> (DeletePhoto)

User -> (CommentActivity)

User -> (FollowUser)

Host -> (EditDeleteActivity)

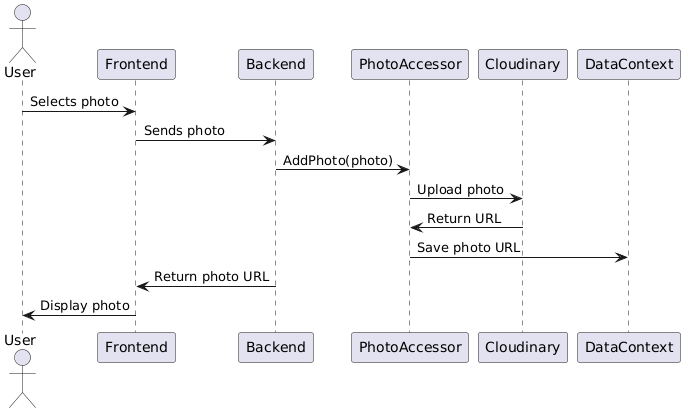
System -> (HandleLogic)

System -> (StoreData)

System -> (ManagePhotos)

@enduml

## Diagram sekwencji (Sequence Diagram)



Kod:

@startuml

actor User

participant Frontend

participant Backend

participant PhotoAccessor

participant Cloudinary

participant DataContext

User -> Frontend : Selects photo

Frontend -> Backend : Sends photo

Backend -> PhotoAccessor : AddPhoto(photo)

PhotoAccessor -> Cloudinary : Upload photo

Cloudinary -> PhotoAccessor : Return URL

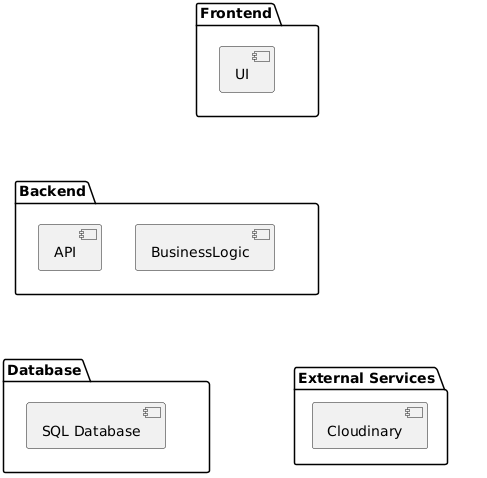
PhotoAccessor -> DataContext : Save photo URL

Backend -> Frontend : Return photo URL

Frontend -> User : Display photo

@enduml

## Diagram komponentów (Component Diagram)



Kod:

@startuml

package "Frontend" {

[UI]

}

package "Backend" {

[API]

[BusinessLogic]

}

package "Database" {

[SQL Database]

}

package "External Services" {

[Cloudinary]

}

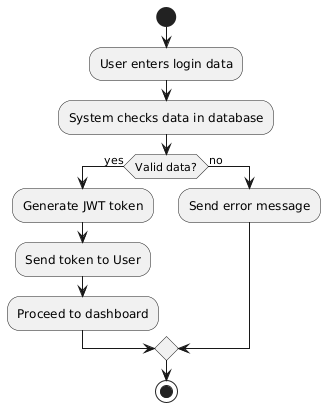
Frontend -[hidden]-> Backend : communicates through API

Backend -[hidden]-> Database : communicates with

Backend -[hidden]-> Cloudinary : communicates with

@enduml

## Diagram aktywności (Activity Diagram)



Kod:

@startuml

start

:User enters login data;

:System checks data in database;

if (Valid data?) then (yes)

:Generate JWT token;

:Send token to User;

:Proceed to dashboard;

else (no)

:Send error message;

endif

stop

@enduml