```
// backend/prisma/schema.prisma

model User {
  id           String   @id @default(uuid())
  username     String   @unique
  email        String   @unique
  passwordHash String
  profilePhoto String?
  bio          String?
  createdAt    DateTime @default(now())

  posts        Post[]
  lists        List[]
  followers    Follow[] @relation("Followers")
  following    Follow[] @relation("Following")
  reactions    Reaction[]
  comments     Comment[]
}

model Post {
  id           String   @id @default(uuid())
  userId       String
  caption      String
  isPublic     Boolean  @default(false)
  locationName String
  locationLat  Float
  locationLng  Float
  price        Float?
  foodType     String
  createdAt    DateTime @default(now())

  user         User     @relation(fields: [userId], references: [id])
  photos       Photo[]
  reactions    Reaction[]
  comments     Comment[]
  listItems    ListItem[]
}

model Photo {
  id           String   @id @default(uuid())
  postId       String
  imageUrl     String
  description  String?
  rating       Int?     // Can be null, or 3/5/10 scale
  order        Int      // Order in carousel
  isFrontCamera Boolean  @default(false)

  post         Post     @relation(fields: [postId], references: [id], onDelete: Cascade)
}
```

```
model List {
  id           String   @id @default(uuid())
  userId       String
  name         String
  description  String?
  ratingSystem Int      // 3, 5, or 10
  isRanked     Boolean  @default(false)
  isPublic     Boolean  @default(false)
  createdAt    DateTime @default(now())

  user         User     @relation(fields: [userId], references: [id])
  items        ListItem[]
}

model ListItem {
  id           String   @id @default(uuid())
  listId       String
  postId       String
  rank         Int?     // If list is ranked

  list         List     @relation(fields: [listId], references: [id], onDelete: Cascade)
  post         Post     @relation(fields: [postId], references: [id])

  @@unique([listId, postId])
}

model Follow {
  id           String   @id @default(uuid())
  followerId   String
  followingId  String
  createdAt    DateTime @default(now())

  follower     User     @relation("Following", fields: [followerId], references: [id])
  following    User     @relation("Followers", fields: [followingId], references: [id])

  @@unique([followerId, followingId])
}

model Reaction {
  id           String   @id @default(uuid())
  postId       String
  userId       String
  type         String   // 'heart', 'thumbsup', 'stareyes', 'jealous', 'dislike'
  createdAt    DateTime @default(now())

  post         Post     @relation(fields: [postId], references: [id], onDelete: Cascade)
  user         User     @relation(fields: [userId], references: [id])

  @@unique([postId, userId, type]) }
```

```
model Comment {
  id           String   @id @default(uuid())
  postId       String
  userId       String
  text         String
  createdAt    DateTime @default(now())

  post         Post     @relation(fields: [postId], references: [id], onDelete: Cascade)
  user         User     @relation(fields: [userId], references: [id])
}
```