# REyt

By Krishna Chennapragada, Tim Mekari, Ali Khan, Brandon Conner

# Functionalities & Features

- Login
  - Username & Password
  - Facebook

- Profile Page
  - Personal account info
  - Profile picture
  - Followers/Following

- Feed Page
  - Pictures from future algorithm
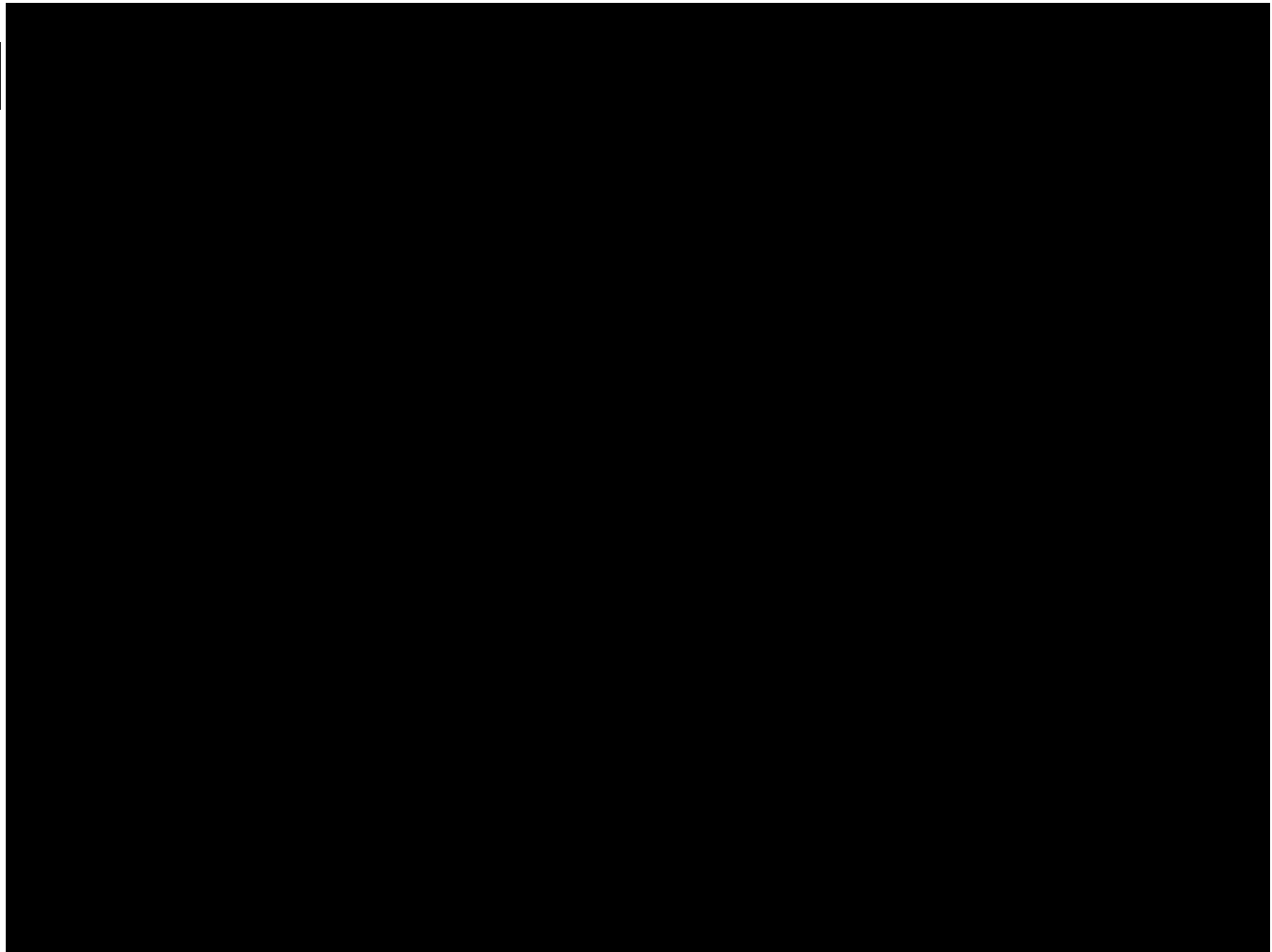
**Functionalities & Features**

- Following/Followers
  - List of users to follow
    - Swipe to follow
  - List of users being followed
  - List of user following you
- Search
  - User list with future application
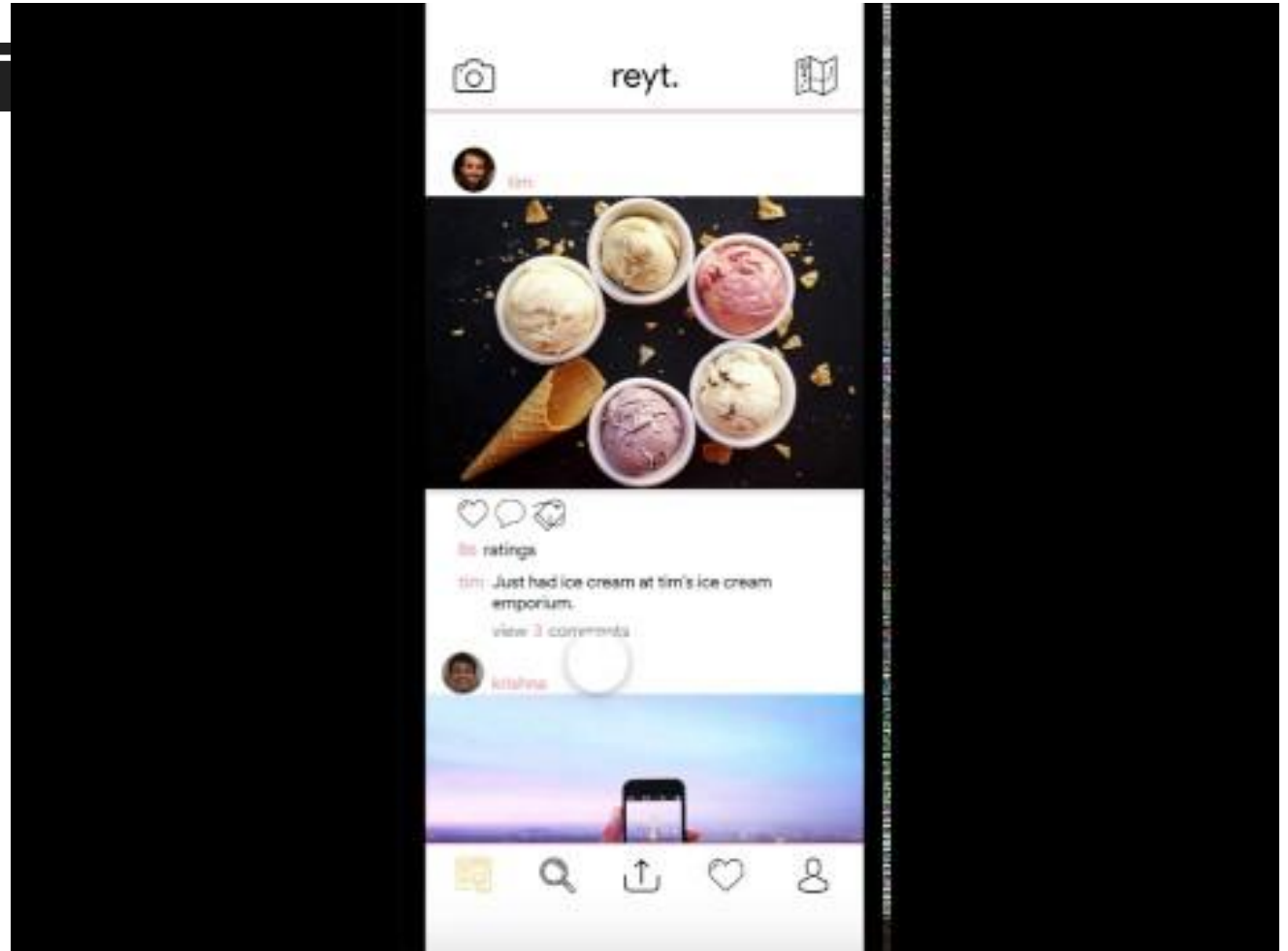
# APP Personality
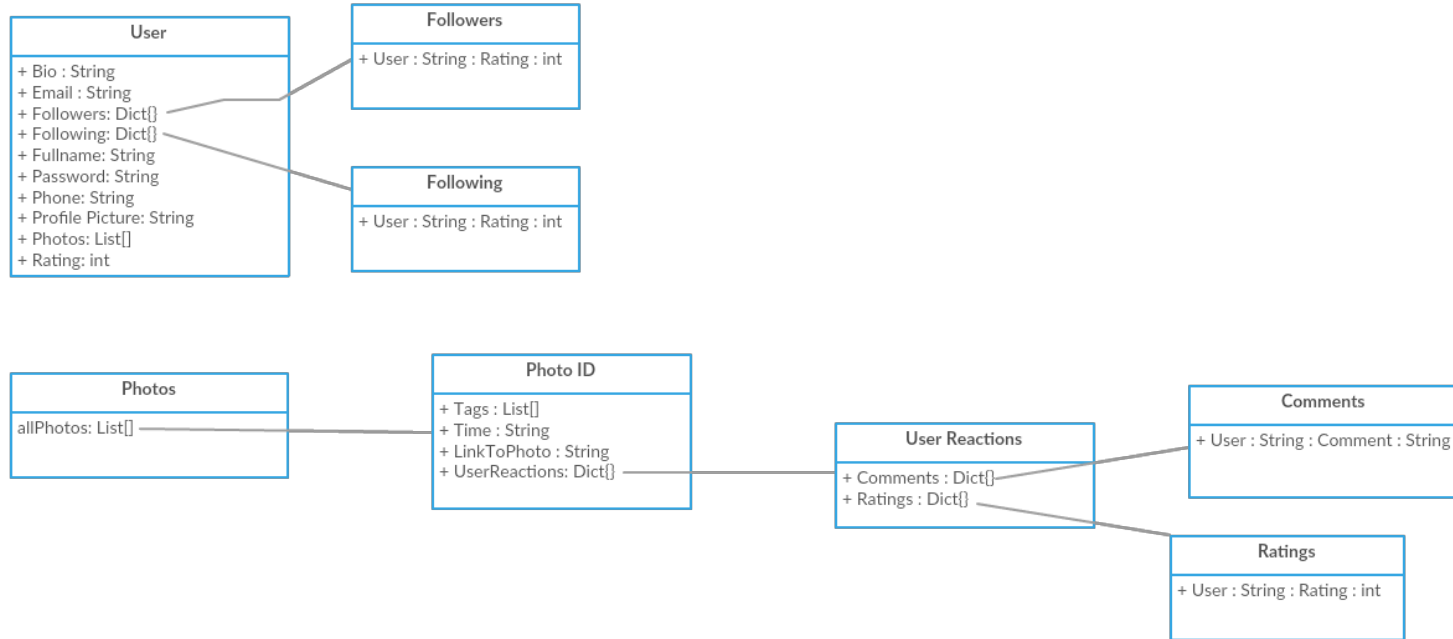




abcdefghijklm
nopqrstuvwxyz

# Low Fidel

# High fideli

# Database SCHEMA

**User**

+ Bio : String
+ Email : String
+ Followers: Dict{}
+ Following: Dict{}
+ Fullname: String
+ Password: String
+ Phone: String
+ Profile Picture: String
+ Photos: List[]
+ Rating: int

**Followers**

+ User : String : Rating : int

**Following**

+ User : String : Rating : int

**Photos**

allPhotos: List[]

**Photo ID**

+ Tags : List[]
+ Time : String
+ LinkToPhoto : String
+ UserReactions: Dict{}

**User Reactions**

+ Comments : Dict{}
+ Ratings : Dict{}

**Comments**

+ User : String : Comment : String

**Ratings**

+ User : String : Rating : int
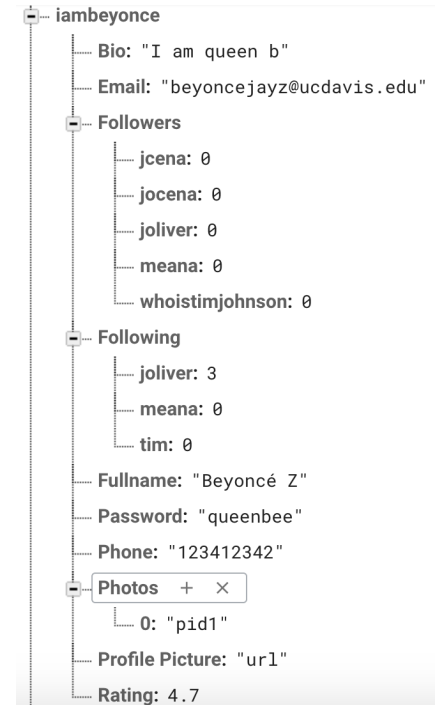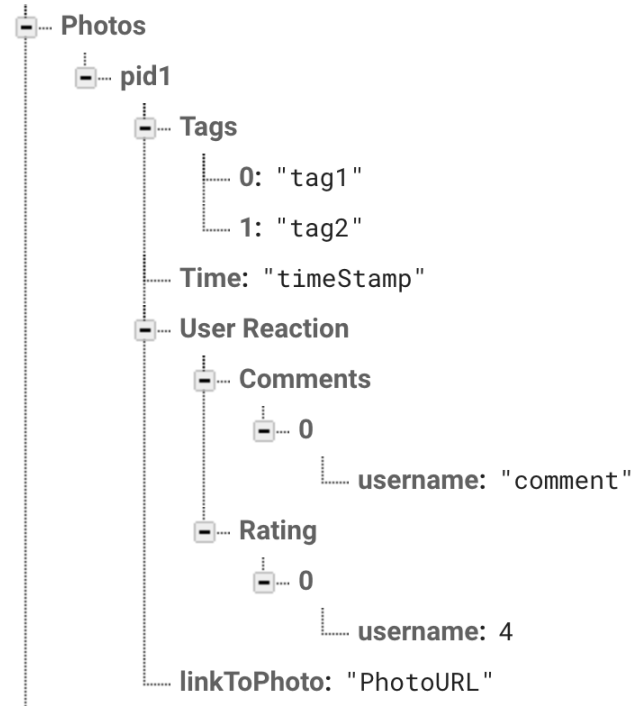
# Database view - user

- The user retains all the regular user details.
- Has a list of following Users and followers.
- Also has a identifier for images posted.
- Each follower also has their ratings for calulations on highest rated users followed.

```
iambeyonce
    Bio: "I am queen b"
    Email: "beyoncejayz@ucdavis.edu"
    Followers
        jcena: 0
        jocena: 0
        joliver: 0
        meana: 0
        whoistimjohnson: 0
    Following
        joliver: 3
        meana: 0
        tim: 0
    Fullname: "Beyoncé Z"
    Password: "queenbee"
    Phone: "123412342"
    Photos  +  ×
        0: "pid1"
    Profile Picture: "url"
    Rating: 4.7
```

# Database view - photos

- Photos have unique ids
- Tags based on related topics
- Time stamp
- User reactions for ratings & comments.
- Link to the photo

```
⊟ Photos
    ⊟ pid1
        ⊟ Tags
            0: "tag1"
            1: "tag2"
        Time: "timeStamp"
    ⊟ User Reaction
        ⊟ Comments
            ⊟ 0
                username: "comment"
        ⊟ Rating
            ⊟ 0
                username: 4
    linkToPhoto: "PhotoURL"
```

# Querying Data

- For querying data within Firebase, we would need to create a reference to our Database within our program.
- We would need to read from and write to children of our application. This was done utilizing Firebase's inbuilt API.

# Queries Used - writing

- **Writing Data:**
- firebaseReference.child("Users").child(username).child("Name").setValue("John Cena")
- Equivalent to: update Users set name= "John Cena" where username=username;

  When we **create an account or update the account or adding new followers** information we query the database to write/update the database.

# Queries used - reading

- **Reading Data:**
- firebaseReference.child("Users").child(users_username).observeSingleEvent(of: .value
  , with: { (snapshot) in
  let value = snapshot.value as? NSDictionary
  value?["Fullname"]...
- **Equivalent to:** select Fullname from Users where username = users_username;

  We query the database to read to retrieve account information when **logging in or displaying user information or accessing the list of users when adding followers. We use this to retrieve all relevant information for each page.**

# Queries used - delete

- **Deleting Data:**
- firebaseReference.child("Users").child(username).child(follower).child(follower _name).removeValue()
- **Equivalent to:** delete from users follower=follower_name where username=username;

  These queries are used to **remove users from a list of following users.**

# Demo!