Android Chat app using Firebase and Material Desig

1. Configuration /configuration

Static client configuration files to so changes can be changed on the fly, if needed

Structure


```
{
 "configuration": {
  "chat": {
   "headers": {
    "string_max": 1000
   },
   "messages": {
    "string_max": 1000
   }
  },
  "matches": {
   "records_page": 20
```

```
  },
  "profiles": {
   "bio": {
    "string_max": 20,
    "string_min": 0
   },
   "images_max": 4,
   "name": {
    "string_max": 20,
    "string_min": 3
   },
   "upload_image": {
    "bytes_max": 3000000,
    "height_max": 1000,
    "length_max": 2000,
    "Storage": "Storage-location",

    "bucket": "your-bucket-id"
   }
  }
 }
}
```

Security Rule

- **Read: Anyone**

- Write: only System Administrator

```
{
 "configuration": {
   ".read": true,
   ".write": false
 }
}
```

2. Profile `/profiles/$user`

Personal information of each user.

Structure

- `$user`: UID of user
- Depends on how the client use `avatar/coverImage` against *profiles*, we can consider to separate them to new tables.
- All images will stored in Firebase storage, under `/storage/your-bucket-id/$user`

```
{
 "profiles": {
   "$user": {
     "public": {
       "name": "User name",
```

```
        "yearOfBirth": "1970",
        "gender": "male",
         "avatarStandard": "$storageStandard",
         "avatarThumb": "$storageThumb",
        "coverImageStandard":"$storageStandard",
        "coverImageThumb": "$storageThumb",
        "bio": "Some information",
        "interests": "some, interesting, things",
        "createdAt": 1473414120,
        "updatedAt": 1473414120
      },


      "private": {
        "dob": "1970-01-01",
        "createdAt": 1473414120,
        "updatedAt": 1473414120,
         "role": "admin",
         "fake":  false,
      }
   }
  }
}
```

Security Rule

- **Read:**
    - **Public: every one**
    - **Private: Owned resource**

- Write
    - Logged in
    - Must owned resource

```
{
  "profiles": {
    "$user": {
      ".write": "auth != null && auth.uid == $user",
      "public": {
        ".read": true
      },
      "private": {
        ".read": "auth != null && auth.uid == $user"
      }
    }
  }
}
```

3. Match Settings

/searchingUsers/$user

Structure

```
{
 "searchingUsers": {
  "$user": {
   "age": {
    "maximum": 28,
    "minimum": 18
   },
   "distance_max": 1000,
   "genders": 3
  }
 }
}
```

* genders 1: man , 2: female, 3: man, female

Security Rule

```
{
 "searchingUsers": {
  ".read": "auth != null && auth.uid == $user",
  ".write": "auth != null && auth.uid == $user"
 }
}
```

4. gridPictures /gridPictures/$user

Structure

- For easy handle number of images, I recommended to use image object keys as number from 1 to LIMIT (5 at the moment)

```
{
 "gridPictures": {
  "$user": {
   "stadandard" : {"0" : " $gridStandard0",
                   "1" : " $gridStandard1",


                   .............
                   "5" : " $gridStandard5",
                   },
 "thumb",  :       {"0" : " $gridThumb0",
                   "1" : " $gridThumb1",
                  ..........
                   "5" : " $gridThumb5",
                   }
   }
```

```
}
```

Security Rule

```
{
 "gridPictures": {
  "$user": {
   ".read": true,
   ".write": "auth != null && auth.uid == $user",
   "$gridImageId": {
    // This approach is must for performance
    ".validate": "$gridImageId > 0 && $gridImageId < 6"
   }
  }
 }
}
```

5. Like `/likes` & Dislikes `/dislikes`

Structure

- History of who liked or disliked

```
{
 "likes": {
  "$like": {
   "userId": "google:123456789",
   "potentialId": "google:987654321",
   "createdAt": 1473414120
  }
 },
 "dislikes": {
  "$dislike": {
   "userId": "google:123456789",
   "potentialId": "google:987654321",
   "createdAt": 1473414120
  }
 }
}
```


Security Rule


```
{
 "likes": {
  ".read": true,
  ".write": "auth != null"
 },
```

```
 "dislikes": {
   ".read": "auth != null && auth.uid == $user",
   ".write": "auth != null && auth.uid == $user"
 }
}
```

6. Reports /reports

Structure

```
{
 "reports": {
   "$user": {
     "$potentialUser": {
           "createdAt": 1473414120

           "reason": “Report reason”
       }
 }
}
```

Security Rule

- Read
    - Logged in
    - Must owned resource

```
{
  "reports": {
    ".read": "auth != null &&
root.child('reports').child($like).hasChild(auth.uid)",
    ".write": "auth != null"
  }
}
```

7. Channel meta data

/channelHeaders/$user/$potentialUser

Structure

- Can access the conversion via REST url
- `createdAt` it shows when chat started

- `updatedAt` it shows when last message sent or received

`/channel-headers/user-uuid-who-sent/user-uuid-who-received`

```
{
 "channelHeaders": {
  "$user": {
     "$potentialUser": {
        "createdAt": 1473414120

        "updatedAt": 4534535434
     }
  }
 }
}
```

Security Rule

- Read
    - `auth.uid` match with `$user` (Sent) or match with `$potentialUser` (Received)
- Write
    - `auth.uid` match with `$user` (Sent) or match with `$potentialUser` (Received)

```
{
 "channelHeaders": {
```

```
    ".write": true,
    "$user": {
      ".write": true,
      "$potentialUser": {
        ".read": "auth != null && (auth.uid == $user || auth.uid ==
$potentialUser)",
        ".write": "auth != null && (auth.uid == $user || auth.uid ==
$potentialUser)"
      }
    }
  }
}
```

8.Conversations/conversations/$user/$potentialUser

Structure

- Can access the conversion via REST url
- "reply":true means message received and if false then its mean message sent
- Conversation created for both users sender will have value of "reply" false and receiver will have value of "reply" true.

- To know if message seen or not then check value of updatedAt if it is greater than zero than its means user opened that message (make sure check that if reply: false).
- `/conversations/user-uuid-who-sent/user-uuid-who-received`

```
{
 "conversations": {
  "$user": {
   "$potentialUser": {
    "$message": {

     "reply": true,
     "message": "I would like to talk with you",
     "createdAt": 1473414120,
     "updatedAt": 1473414120
    }
   }
  }
 }
}
```

Security Rule

- Read
  - `auth.uid` match with `$user` (Sent) or match with `$potentialUser` (Received)
- Write

- `auth.uid` match with `$user` (Sent) or match with `$potentialUser` (Received)

```
{
  "conversations": {
    ".write": true,
    "$user": {
      ".write": true,
      "$potentialUser": {
        ".read": "auth != null && (auth.uid == $user || auth.uid == $potentialUser)",
        ".write": "auth != null && (auth.uid == $user || auth.uid == $potentialUser)"
      }
    }
  }
}
```

11. User location

/locations/$user

Standard Structure of geofire

```
{
 "locations": {
   "$user": {

     "g": “geofireIndex”
     "l": {
        0: 123.4567,
        1: 765.4321
     }
   }
 }
}
```

Security Rule

```
{
   "location": {
      "$user": {
         ".read": false,
         ".write": "auth != null && auth.uid == $user"
      }
   }
}
```

12. Fake Users /fakeUsers

Structure

```
{
 "fakeUsers": {
  "$fakeUser": {
     "createdAt": 1478635732020,
     "createdBy": "i7prrf0a8HU7FSivI4ea1lggDFe2"
  }
 }
}
```

13. Channel Likes /channelLikes

Structure

- List of liked and disliked users
- `true` potentialUser liked and `false` potentialUser disliked

```
{
  "channelLikes": {
    "$user": {
      "$potentialUser1": true,
      "$potentialUser2": false
    }
  }
}
```

14. Storage

- **Maximum five standard and five thumb gridPictures**

gridPictures**{**

"$user": {

$timestamp_standard.jpg,

$timestamp_thumb.jpg,

}

}

avatars**{**

"$user": {

avatar_standard.jpg,

avatar_thumb.jpg,

}

}

```
covers{

        "$user": {

                    cover_image_standard.jpg,

                    cover_image_thumb.jpg

        }

}
```