My Community Notification System Changelog

This Document will guide you to implement the new Notification System using Firebase.

Notifications Type :

at first let's take a look at what notification types we have

```
enum  NotificationType  {
    CHAT_MESSAGE  =  1000,
    STATUS_LIKE,
    STATUS_DISLIKE,
    STATUS_SHARE,
    STATUS_REPLY,
    CONTACT_NEW_STATUS,
    CONTACT_LIVE_VIDEO,
}
```

let's discuss every type in brief

- `CHAT_MESSAGE`
  This type of notifications should be handled in the chat system
  but it is good to know about it.

  - What Payload do you expect ?

```
data:  {
    type: 1000,
    hasMedia: if the message have media,
    content: Message Content,
    senderId,
    conversationId,
  },
  notification:  {
    body: Message Content!,
    title:  Sender First Name,
    icon:  Sender Profile Pic,
    type:  1000,
  },
```

  - When I will receive this notification ?
    You will get this notification if the user was offline and someone sends a message.

- `STATUS_LIKE,STATUS_DISLIKE,STATUS_SHARE,STATUS_REPLY`
  all these types considered the same, in the shape of payload, but as you might noticed that every type correspond to status action type.

  - What Payload do you expect ?

```
notification:  {
    statusType: Media,Status,Story...,
    title:  Sender First Name,
    icon:  Sender Profile Pic,
    type:  1001...1004,
  },
data:  {
    statusId,
    type:  1001...1004,
    hasMedia,
    senderId,
    statusOwner: The Id of the Status Owner,
    statusType: Media,Status,Story...,
},
```

  - When I will receive this notification ?
    You will get this notification if someone reacted on one of your statuses or status you follow.
  - How do I follow a status ?
    You will be auto added to status followers once you reacted on that status, but you can also follow the status without creating any action on it, we will discuses this later.

- `CONTACT_NEW_STATUS`
  this an important notification type, you will git notified when some of your friends create a new status.

  - What Payload do you expect ?

```
data:  {
    statusId,
    type:  1005,
    hasMedia,
    statusOwner: The Id of the Status Owner,
    statusType: Media,Status,Story...,
  },
  notification:  {
    body: Status Content!,
    statusType: Media,Status,Story...,
    title:  Sender First Name,
    icon:  Sender Profile Pic,
    type:  1005,
  },
```

   - When I will receive this notification ?
     You will get this notification if someone of your friends published a new status of any type

- `CONTACT_LIVE_VIDEO`
  this notification type is the same as above except one new property `channelId`

   - What Payload do you expect ?

```
data:  {
    statusId,
    type:  1006,
    hasMedia,
    statusOwner: The Id of the Status Owner,
    channelId: The Live Stream Channel Id
    statusType: Media,Status,Story...,
  },
  notification:  {
    body: Status Content!,
    statusType: Media,Status,Story...,
    title:  Sender First Name,
    icon:  Sender Profile Pic,
    channelId: The Live Stream Channel Id
    type:  1006,
  },
```

   - When I will receive this notification ?
     You will get this notification if someone of your friends create a live video stream.

Notes:

1. Data Types:
   Note that every data type in the payload of notification is a string.
   this limitation from Firebase, i don't know why !
   anyway be careful !!
2. Notification Latency:
   Firebase may schedule the notification for some reasons like battery power saving.
3. you should handle notification icon and title if it was missing, since one user may not have a profile pic so the value will be `"null"`.

Subscribing to an Entity :

in this section we will talk about subscribing/unsubscribing to/from `User` or `Status`

but first you need to provide your `deviceToken`, WHAAT !
a Device what ?
ok, the `deviceToken` is just an abstraction concept around the `firebase token` so, you need to send your firebase token.

> see API Documentations at section `Users` at endpoint `/user/update/device-token`.

- Turning on/off notifications:
  The user can turning on/off the notification for a particle user or a status at any time, so of course there is an endpoints to do that.

  - `/notifications/subscribe`
  - `/notifications/unsubscribe`

> NOTE: you need to be `Authenticated` to use these endpoints,
> for more informations about the payload you should provide see the API Documentations.

- Syncing User Notifications:
  Since the user notifications is cached in the application, we need to provide a way to get this notifications back if the user deleted the app/cache.
  The Problem: "I DON'T HAVE TO TRACK THE USER SUBSCRIPTIONS"
  yes, that's the problem, since we relay on `Firebase` so I used one of powerful firebase future is Notifications Topics , I don't need anymore to track the user subscriptions, i just will send to firebase the topic name and the user token, and

let the rest on firebase.

when something new happened like someone of my contacts published a new status, since i’m subscribed to that user by default when i added him to my contacts, the server will publish to that user topic that he published a new status, since all of his followers subscribed to that topic, firebase will send that notification to all of his followers automatically.

We can sync user notifications, by just starting to keep tracking of his subscriptions - which it is a bit strenuous - and then save all his notifications. but what i’m thinking about it, we can make a good solution for that problem: "I WILL NOT RECORD THE USER NOTIFICATION UNTIL HE RECEIVES IT"

but how ? simply, when you receive that notification we will build a new payload of it and send it back to the server, that way we can get rid of the hard way of keep tracking of all the user subscriptions and let the user take care of it.

Take a note that not a final decision of syncing the user notifications, we need to discuss this later.

Any Questions ?