NAME: SAMEEDD TARIQ, SALMAN SHAHID, WAJIHULLAH, Abu Huraira

ROLL NO: FA19-BSE-(029,013,115, 041)

**Introduction to Instagram:**

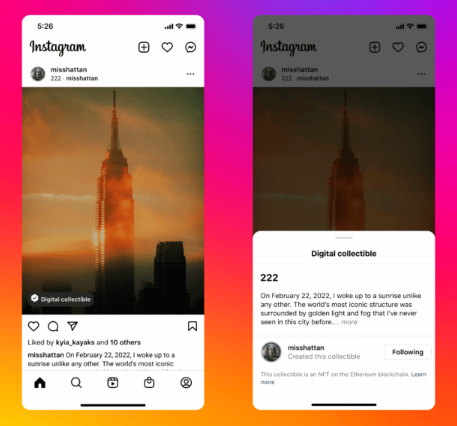
With so many Instagram upgrades, it's difficult to even consider it as a single, consistent app because Instagram has always been changing. In fact, there have been so many changes over the years that discussing the numerous "Instagrams" that have been introduced since 2010 would be more appropriate.

Today, Instagram has solidified its place alongside Facebook, YouTube, and Twitter as one of the major social media platforms. Instagram has demonstrated that it is here to stay with the introduction of Instagram Reels just a few months ago.

**FEATURES:**

**August 2022**

In August 2022, Instagram announced that the platform would start supporting its NFT feature in 100 countries, including the US, Africa, Asia-Pacific, the Middle East, and the Americas. Instagram users will be able to share NFTs on the Ethereum, Polygon, and Flow blockchains by connecting their digital wallets to the application. On Instagram, sharing NFTs is completely free. It is completely free.



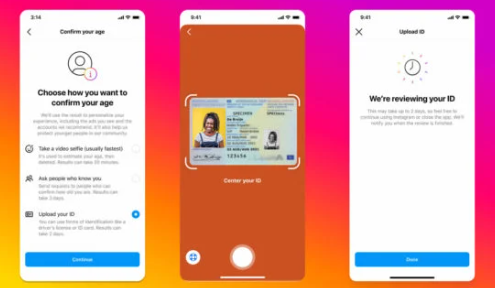
**July 2022**

With the new Dual Features and New Instagram Reels Templates that were revealed in July, Instagram is pushing everyone to make reels and takes reels extremely seriously. Users may make Reels very quickly thanks to these pre-made templates. Even if some people are unaware of how Reels operate, creating a Reel is really simple and quick with just a click of the "Use template" button. Users may now record material and responses concurrently on Instagram thanks to the new Dual functionality. You may add a different viewpoint to your Reels utilising the front and back cameras on your phone by using the Dual function in the Instagram camera.



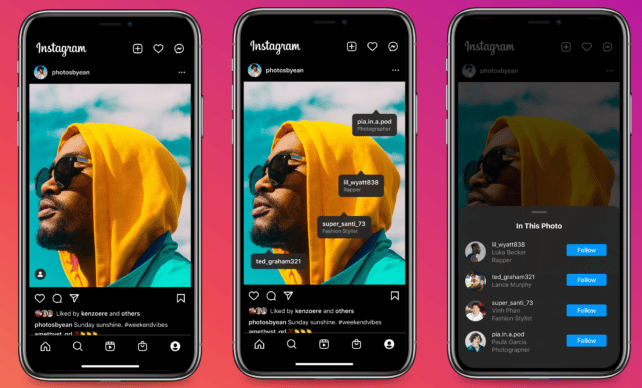
**June 2022**

New ways to confirm your age



**March 2022**

Enhanced tags, In order to give artists credit, Instagram launched the Enhanced Tags feature. This feature enables users to tag all contributors and collaborators in a post, enabling you to give credit to everyone who was engaged in the creation of the content, even if they were only a minor contributor (photographers, makeup artists, designers, etc). This makes it simpler for these individuals to gain recognition and discover fresh partnerships and chances.



**January 2022**

Chronological order of feed, Instagram experimented with changing the feed and provided three new options: Home, Favorites, and Following, which put posts in a more chronological order. The sequence in which posts show in a user's feed can be customised.

Instagram's algorithmic feed serves as its "home," however it has been noted that there may be some modifications, such as a rise in the amount of suggested posts. The much-anticipated new Instagram chronological feed is presented after. By selecting this, users are able to view content just from the profiles they are following, naturally in chronological order. Users can manually add their favourite persons to the Favorites feed so that they can only view material from them.

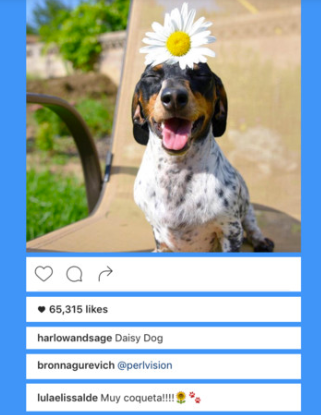
**Refactoring at scale – at Lessons of rewriting Instagram:**

**Technical Debt:**

Despite being 22 years old, Instagram still uses the same code foundation. There are commits for the Instagram init code there if you look through the git history and blame a few files. There is a lot of chaos and there are still a few pockets of manual memory management. There were other factors that made it quite difficult.

**Working:**

We begin by using collection views. When viewing a post on Instagram, we divide the large section into numerous smaller cells. This breakdown functions as follows:



You have a supplementary view at the top, media cells, cells for all of the action items, cells for all of the text cells, and cells for all of the action items. A data model that we refer to as a "feed item" is what powers this.

You only have to look at one feed item to see how many comments there are, whether or not to display an image or video, who the person is, etc. The feed item data model serves as the foundation for our entire software, which anticipates that each feed item will contain an image, a video, comments, etc.

When Instagram first began in 2010, all it had were photographs. Over time, newcomers have arrived who wished to include things like video, users, and various data models. We've all probably wanted to make a small adjustment, but instead of refactoring and acting morally, we just wanted to slap it on.

That's what we did, actually. We had a huge model instead of all these extra little enclosed models. It got quite challenging and started to slow us down. Now remember this section cell mapping is driven by a feed item. If you’re looking at a feeds in Instagram, you’re not looking at one post: you’re seeing a bunch.

Something must take this data model, section it off, and set up all of these cells. It was our view controllers' job to take care of that. (Yes, the plural form of controllers.) We had a view controller for our collection view, which was derived from which we had a view controller for our networking, which was derived from which we had a view controller for general feeds, and from which we had a view controller for the main feed tab of the app. spanning four storeys. A new cell proved difficult to add.

**Feed 2.0:**

Instagram’s main objectives were to address the issue of view controller inheritance, simplify the feed, and enable the usage of various cell types and data models. The "feed item" concept, which was completely independent of our data types, was something we wanted to do away of.

**Diffing:**

Instagram began with diffing, which is a notion that involves having a variety of items, such as a collection of models. When you visit another array in this place, its values update because items have been removed, added, or moved about. When creating infrastructure, diffing is quite useful, but it can be challenging to use collection view correctly.

Prior to reloading, moving anything into its final location, and then doing inserts based on the final index, you must first delete items from the old array. It requires a little bit of math to execute it correctly.

The majority of simplistic diffing implementations are n2 in complexity. It can be slow while performing that many processes. The majority of implementations I've seen go to a background queue, perform the calculations, then return to the forefront and continue, but even that is a bit sluggish. You have a lower priority queue working on a more challenging issue. Instead of using the main thread, why not?

We looked around and came across a paper by a man named Paul Heckel that was written in 1978. The least common subsequence, or LCS, was used in this study to solve the problem in linear time.

By doing this, we were able to develop an algorithm that will quickly identify all deletes, reloads, inserts, and movements between two sets of data. It will allow us to do it on the primary queue, allowing us to do all of these adjustments on the collection view, which is a lot more straightforward model for us. Furthermore, it took a lot longer than you might imagine for us to understand how collection view functions.

**Applying Updates:**

Regarding the clutter created by view controllers, we cleaned up a lot of garbage and moved it into shared objects, systems, libraries, etc. Not all of them have to be view controllers. The fundamentals of networking, such as analytics, can all be contained in one object. However, we were still faced with the feed.

This made use of a notion we refer to as "The World," where the view controller was aware of the array of objects, how they are organised into sections, how those sections are set up, and how the cells are filled. It handles everything related to interaction, logging, display events, and other things.

**Item Controller:**

We made the decision to divide things up in the new infrastructure that we built. An "item controller" is an abstraction we developed. Simply put, it's a tiny view controller for a section.

The amount of items, cell configuration, cell size return, and interaction management are all done here. However, it's most crucially where you keep all of your business logic. Even though it's a simple collection view, the way we've divided things up makes it possible for us to include any other kind of object in our collection views.

Simply build a new item controller, and it will take care of everything else.

Though we initially believed it to be impossible, we succeeded. The group was content, so we shipped this.

**IGListKit:**

Instagram is releasing IGListKit, a brand-new framework that handles all of this for you (release tba).

Instagram will only use Objective-C nullability, annotations, and generics in the 100% Swift code that makes up all of our sample apps and documentation. This is completely compatible with Swift, and the C++ is hidden so deep you'll never have to look at it.

**IGItemController:**

IGItemController is one of the most significant classes in this framework. It is the "item controller" idea that I alluded to earlier. There isn't a lot of code here. This only affects one cell with a text label. I'm done now.