Instagram

Alternative Recommendation System

IN4315: Software Architecture

Irtaza Hashmi
Kevin Nanhekhan
Xinrui Xu
Zenan Guan

Delft University of Technology

TU Delft

Instagram

Alternative Recommendation System

by

Irtaza Hashmi
Kevin Nanhekhan
Xinrui Xu
Zenan Guan

Student Name	Student Number
Irtaza Hashmi	4829360
Kevin Nanhekhan	4959094
Xinrui Xu	5715393
Zenan Guan	4813855

Word count:	7904
Instructor:	Arie van Deursen and Diomidis Spinellis
Teaching Assistant:	Boriss Bērmans
Project Duration:	February, 2023 - April, 2023
Faculty:	Faculty of Electrical Engineering, Mathematics and Computer Science
GitLab repository:	`https://gitlab.ewi.tudelft.nl/in4315/2022-2023/sa-team20`

TU Delft

Contents

1

Conclusion

Instagram is one of the most popular social media platforms where users share photos and videos, as well as engage with other users by liking, commenting, and messaging. It was launched in 2010 and is owned by Meta. Instagram's recommendation system is a crucial part of its system that keeps its business running.

Analysis Conclusion: There is no doubt Instagram's current recommendation system is state of the art and keeps the users glued to their screens, however, our analysis has brought up two main problems that need to be addressed:

1. The current recommendation system of Instagram has a system-centred design instead of a user-centred design. (See Section 3)
2. The current recommendation system of Instagram is a black box and lacks transparency. (See Section 3)

Actionable Advice The team proposes a new hybrid recommendation system architecture that allows users to:

1. Select their own recommendation system according to their own choice. (See Section 4)
2. Understand the working of the recommendation system to increase transparency. (See Section 4)

The rest of the report is structured as follows. In Section 2, we analyze the product and define the design problem of why Instagram needs to change its current recommendation system. Section 3 presents the functional and non-functional requirements and gives an overview of the current architecture using the C4 model. Next, Section 4 proposes alternative recommendation system architectures for Instagram with their respective quality attributes, stakeholders, and tradeoffs. Proof-of-concept is discussed in Section 5 with its quality assurance discussed in Section 6. Finally, further improvements that can be made to the proof-of-concept and the experiments are suggested in Section 7.

2

Product Analysis and Design Problem

It is important to focus on Instagram in general in terms of what its capabilities and use case are, identify the stakeholders of the system, consider the ethical aspects, and lastly describe the design problem of the system.

2.1. Main Capabilities & Use Case

The main capabilities of Instagram include:

Sharing photos and videos: The most important function of Instagram is to allow users to share photos and (short) videos, alongside allowing users to edit their photos and videos through various tools such as filters.

Following other users: Instagram allows users to follow other users and browse their content.

Liking and commenting: Instagram allows users to like photos and videos of other users and comment on them. This user engagement is then used for the recommendation system which recommends users content they may be interested in.

Direct messaging: Although not the most featured function, Instagram allows users to send instant messages to other users or create group chats.

Explore page: Instagram allows users to explore content and/or content providers they potentially may like.

2.2. Stakeholders

The stakeholders we have identified are listed below. Also, their influence and interest are analysed using the stakeholder matrix shown in Figure 2.1 below.

Users: Users are the people who use Instagram to share their daily life, discover interesting content and interact with others on the platform.

Instagram business: The Instagram business is the Instagram company itself. It is responsible to run the Instagram business, report to the investors, and devise new business plans according to the market and the feedback from users

Influencers: Influencers are those who have a lot of followers on the platform. They are the main content providers of Instagram.

Advertisement providers: Advertisement providers are those who do business and pay Instagram to have their ads displayed to the users in order to promote their goods/services.

Investors: Instagram's investors, including their parent company (Meta) and other stockholders, provide funding for maintaining the platform, and they care about Instagram's financial performance.

The Engineering Team: The engineering team of Instagram is responsible for developing and maintaining the application as well as updating and adding new functions according to the requirements of the Instagram business.

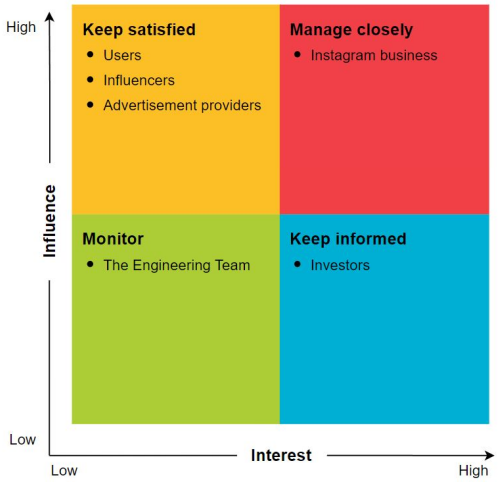

Figure 2.1: Stakeholder matrix

2.3. Ethical Considerations

As one of the most popular social media platforms, Instagram should consider some ethical concerns. Here we will discuss some ethical considerations:

1. It's Instagram's responsibility to protect user privacy and data security. Instagram collects a huge amount of user data including their interests and browsing histories which will be used by advertisers to promote products. This will raise ethical concerns about the users' authorization of the data. To address the concerns, Instagram has to with strict data security protocols, and use techniques to ensure data security.

2. Instagram should try its best to ensure the authenticity of the content. Some influencers may spread fake news on the platform to increase followers. Also, some advertisers pay influencers to promote or even exaggerate products, which will cause users to have no ideas which are genuine recommendations and paid advertisements. Instagram should implement clear policies or collaborate with some independent fact-checking organizations to deal with the concerns.

3. Instagram is supposed to prevent cyberbullying. Billions of people are using Instagram. By hiding behind anonymity, some people may harass or attack other users which may cause serious mental issues for other users. Instagram should implement effective reporting systems and moderation systems to prevent this behaviour.

4. Instagram should have a mechanism to detect and freeze bot accounts. Scammers or harassers use bot accounts to lure ordinary users into clicking and redirecting to malicious websites, harass specific users, and spread fake, machine-generated information to defame their reputations.

2.4. Design Problem

Instagram's user base has been increasing over the years. Hundreds of recommendations are generated for each user every day and impact people's lives. Currently, the recommendation system architecture of Instagram is a black box and lacks transparency. Last year, Instagram's algorithm was officially listed as the cause of death by a coroner in a 14-year-old girl's suicide case in the UK [7]. According to the reports, the recommendation system kept showing images of self-harm to the girl. There was no way to ”reset” it. People are not able to control nor understand the recommendation systems which may cause many problems in the future. We would like to change and propose an alternative recommendation system that is user-centric and provides transparency. We propose the user may select what type of recommendation system is used on them and allow them to make their own choices.

3

Main Architectural Style

Having considered Instagram in general, it is also important to look at it from a more technical perspective by analyzing its architecture. Although not every bit of information is available to the general public, we attempted to analyze several aspects from what is available online (literature, blog posts, etc.) and also from elicited requirements.

3.1. Requirements

There are two types of requirements, functional and non-functional requirements. Functional requirements refer to the features of a software system that it must perform to satisfy the needs and expectations of its users. Non-functional requirements, describe the characteristics or qualities that the system must possess to meet its functional requirements. The functional and non-functional requirements of Instagram are given below.

3.1.1. Functional Requirements

As a social app, Instagram must have to fulfill its primary objectives. The functional requirements of Instagram according to the MoSCoW method are:

Must Have
1. Enable users to create an account, log in and manage their profile securely.
2. Enable users to upload and share their photos and videos.
3. Allow users to do social interactions, such as liking, commenting, and sharing posts.
4. Provide users with personalized content from accounts they follow.

Should Have
5. Searching, following, and messaging other users.
6. Notify users of relevant activities such as likes comments and mentions.

Could Have
7. Allow advertisers to promote their products and services to the users.

3.1.2. Non-Functional Requirements

Besides functional requirements, Instagram should have some characteristics to ensure user satisfaction. The non-functional requirements of Instagram according to the MoSCoW method are:

Must Have
1. Scalability: Allows the application to grow with an increasing number of users.
2. Availability: Accessible to users at all times, without experiencing significant downtime.
3. Reliability: Ensure it consistently provides accurate and reliable information.
4. Performance: Ensure that the system loads quickly and performs well under high traffic.

Should Have
5. Usability: Make the interface more user-friendly and easy to interact with.

3.2. External dependencies

The recommendation of Instagram relies on some external dependencies to provide users with personalized content.

1. Machine learning libraries: Instagram uses machine learning techniques to generate recommendations [17]. Some-third party libraries of machine learning such as PyTorch or Tensorflow are necessary for the implementation of machine learning algorithms.

2. Data storage: Instagram uses a variety of cloud-based data storage solutions to store vast amounts of user data to generate accurate recommendations. Instagram primarily relies on Amazon Web Services(AWS) [13] for its cloud-based data storage. AWS, launched by Amazon.com, is a computing platform that provides a wide range of services such as Amazon Elastic Compute Cloud, Amazon Simple Storage Service, and Amazon Relational Database Service.

3. Third-party data providers: Instagram may collect user data provided by third-party data providers. These providers may provide something like users' purchase history, and browsing history which can be used by the recommendation system to generate accurate recommendations.

3.3. C4 Models

3.3.1. Context view

The context view is the first layer of the C4 model. In the context of Instagram, it contains two types of actors: users and advertisers. Users use the platform to post images and videos and comment, like, and discover content from others. Advertisers use Instagram to promote their products and services. The views also include some existing external systems which Instagram relies on, such as the payment system and storage system. Instagram uses the payment system to charge their users who need better service. External storage system, such as Amazon Web Service, is used to store user data.

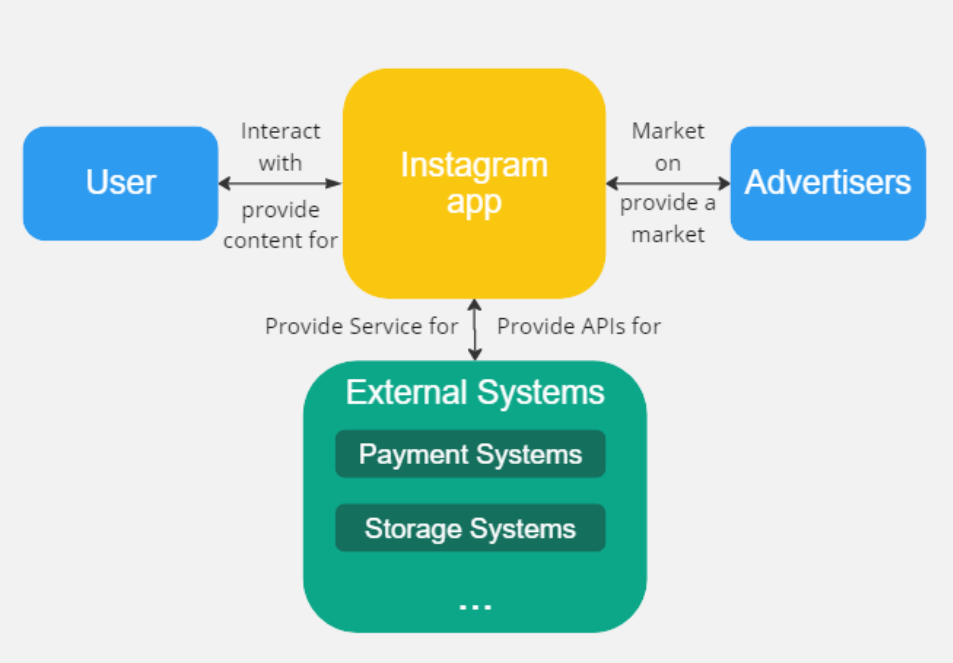

Figure 3.1: Context View

3.3.2. Cotainer view

The container view is the second layer of the C4 model. It shows a high-level software architecture of Instagram and how the responsibilities in the software system are distributed, i.e. among each container.

A Container is a separately runnable unit. For Instagram, the main software structure has four containers: Web application, Mobile Application, API Application, and Database; User can use a browser to use the web application or mobile application to log in and enjoy Instagram's functions, the requests from the client application will be handled by API Application and authentication information will be stored in the database of Instagram.

Some external containers, Notification systems, Email systems, and Payment System and Storage systems, are used to support the functioning of Instagram, where Notification service helps to push notifications to users' mobile phones or web browsers, email system is responsible to send emails to users (for example, password reset), Payment system handles the financial transactions and Storage systems deals with storing the contents (photos, videos, etc) which are public data and costs a lot of storage.

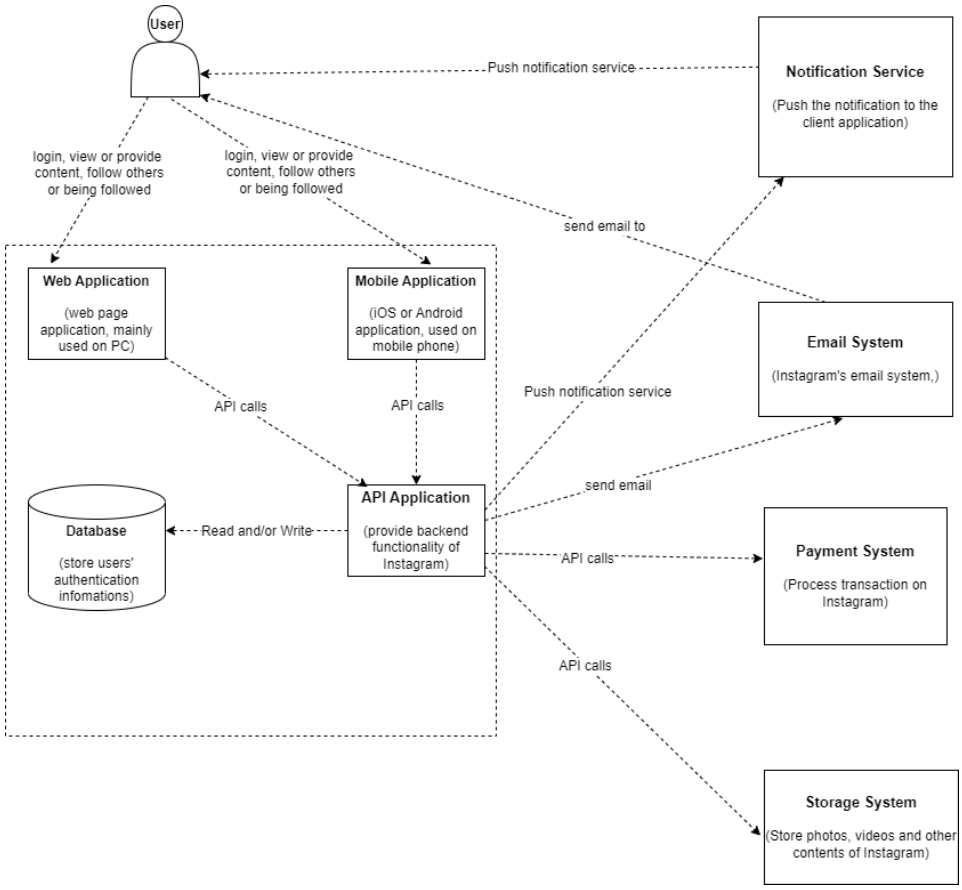

Figure 3.2: Container View

3.3.3. Component view

The component view is the third layer of the C4 model. It shows the components in the containers and shows a deeper insight into the software structure. The requests from the client-side application (web, mobile) will be accepted by different handlers. Then, the components, which are decomposed from containers, will handle the requests and refer to corresponding external systems or databases for further operations.

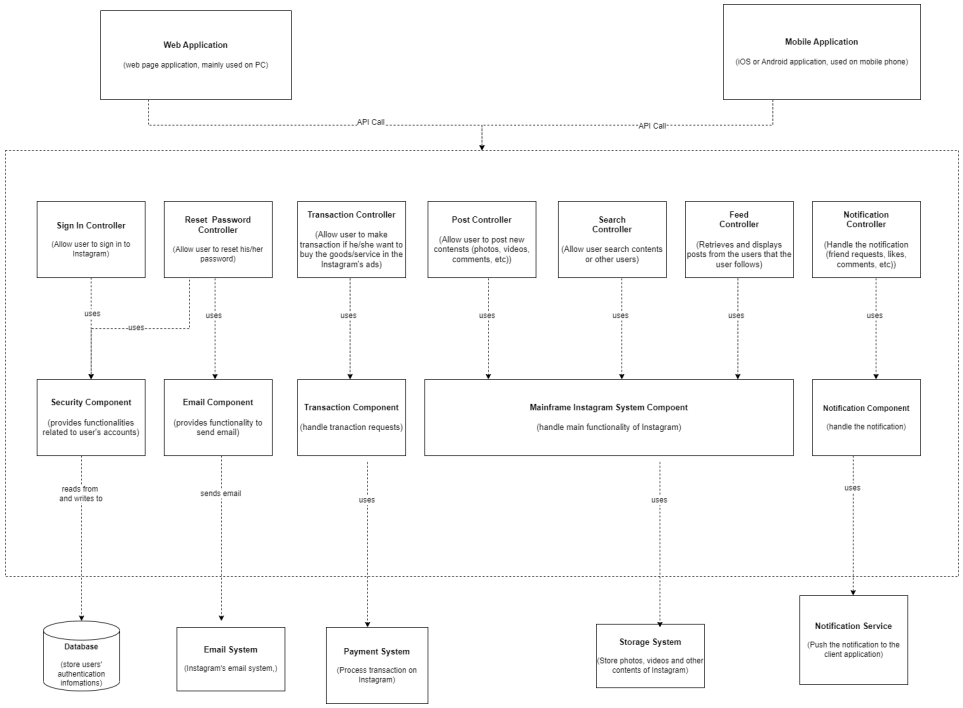

Figure 3.3: Component View

3.3.4. Class view

The class view is the fourth and final layer of the C4 model. The Figure 3.4 shows the class view of the recommendation system of Instagram and models how their recommendation system classes interact with each other.

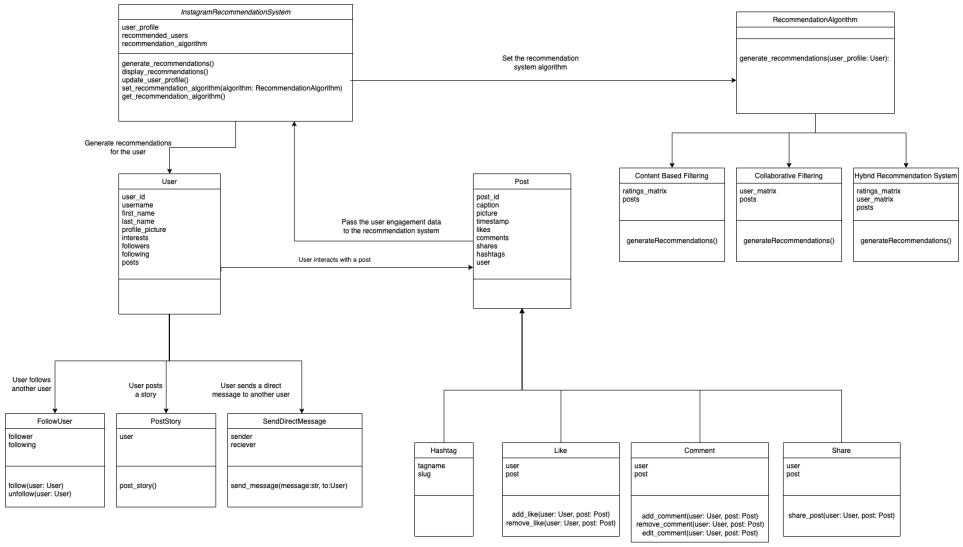

Figure 3.4: Class View

3.4. Key scenarios

User sign up: A user can create an Instagram account through email address, phone number, or Facebook information.

Uploading photos and videos: A user can upload photos and videos from his/her device or shoot a

picture from the app. He/she can also add titles, labels, or positions to his/her posts, as well as delete his/her post.

(Un)following other users: A user can (un)follow other Instagram users, so he/she can see their posts in the timeline. The user can also search for other users or content through keywords.

Liking and commenting: A user can like and comment on other users' posts, to show appreciation or engage in conversations with them.

Saving posts: A user can save a post he/she finds interesting so he/she can find it easily if he/she wants to review it.

Direct messaging: A user can send direct messages to other users. Exploration page: Instagram provides an exploration page to let a user find new content based on their interests and prior activities.

3.5. Recommendation system architecture

The general recommendation system architecture is shown Figure 3.5. We have a database of posts that are passed to the recommendation system along with the user profiles. The recommendation system then generates recommendations and displays them on the Instagram user feed. A user interacts with the posts, which then updates the user profiles. This keeps repeating and the user profiles keep getting updated for better recommendations.

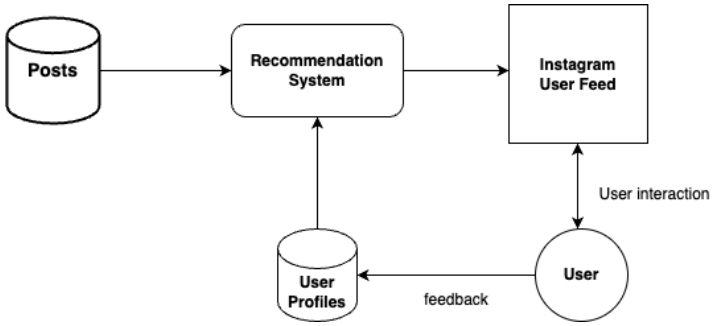

Figure 3.5: Recommendation system architecture

3.5.1. Key domain concepts

Instagram has several pages which use the recommendation system to provide users with personalized content, including a home feed, explore page, and reel where users can see posts of accounts they follow, discover new content, and see short-form videos based on their interests and activity on the platform. A combination of machine learning models [8] and techniques of natural language processing are used in these pages, such as deep neural networks. These models and algorithms will be updated constantly while the user data is being connected and analyzed.

3.5.2. Quality attributes of the system

Security: When recommending content and products to users, the recommendation system will access users' personal information. The user's personal information and browsing history are sensitive data that should be kept from unauthorized use. So security plays an important role in the recommendation system. Instagram primarily uses data encryption to protect user data and publish privacy policies that explain how data is collected and used.

Scalability: Scalability is a necessary quality attribute for the recommendation system. For each user, the recommendation system needs to be personalized. The system must be able to handle large amounts of user data. Instagram has to guarantee the refresh speed of the explore page where users can discover new content. The more frequently the explore page refreshes, the more data the system

has to process.

Performance: Performance plays a very important role in the recommendation system of Instagram. The recommendation system is designed to generate recommendations quickly and ensure users have a smooth experience with little latency. The generated content is also supposed to be accurate, thus enhancing user satisfaction.

3.5.3. Trade-Offs

There are some tradeoffs considering the current system of Instagram. Although the recommendation system of Instagram can provide personalized content to users, this may cause an information cocoon in which users are only exposed to content that reinforces their existing beliefs and opinions, thus leading to a lack of diversity of content. The recommendation system of Instagram has to achieve a balance between personalized content and diversity.

Also, to provide a personalized recommendation, the system will need access to user data. However, in the process of collecting user data, there is a risk of leaking data. Thus, the recommendation system of Instagram needs to protect user data by using certain techniques such as encryption and differential privacy.

4

Alternative recommendation systems

This section will explain the alternative recommendation systems along with their quality attributes and trade-offs from section 4.1-4.4. These alternative recommendation systems include content-based filtering, collaborative filtering, context-aware recommendation system, and hybrid recommendation systems. The user will be able to choose one of the recommendation systems as their preference including a hybrid one with custom weights, which determine the influence of each recommendation system.

4.1. Content-based filtering

Content-based filtering is a recommendation technique that involves using a user's former choices to calculate similarity counts to recommend similar content [16]. This is visually shown in Figure 4.1 below. Most systems rely on two manners to keep track of a user's former choices, namely, by keeping an interaction log of the user with the system and modelling the user's preference. Here the user's profile is modelled at the hand of a weighted vector of content features. The representation of this data is most often done through TF-IDF (Term Frequency-Inverse Document Frequency) to create a sparse vector space representation of the content profiles. By inferring the content attributes and features from the user's preferences and comparing them to other content profiles, a recommendation can be made to fulfil a user's interest.

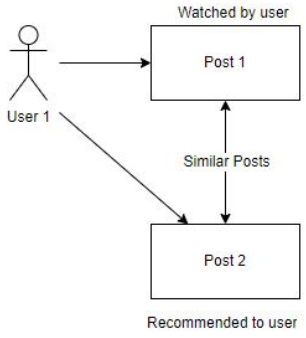

Figure 4.1: Content-based filtering

For example when looking at the above utility matrix (see Table 4.1), we see Alice has positively interacted with content tagged with photography before, but not so well with *#food*. Alice has not interacted with *#picoftheday*. Therefore the system can recommend Alice's content with posts that are related to *#photography*.

	#photography	*#food*	*#picoftheday*
Alice	5	3	-
Bob	4	5	4
Charlie	-	-	3

Table 4.1: Example utility matrix containing interaction information for content-based filtering

4.1.1. Quality attributes of the system

Scalability: A key quality attribute of the content-based filtering recommendation system is that the system does not require other user profiles to make recommendations. This makes scaling the system very easy as only the features of the posted content on Instagram are required to properly model a user's profile.

Security: Compared to the current main recommendation system, the content-based one can be even more secure due to having no need to make comparisons between user profiles for the recommendations made. For example, the current system relies on not only looking at users' interests but maybe also other implicitly gathered features such as location.

Flexibility: Content-based filtering recommendation systems can adjust themselves well in terms of new types of content to recommend to its user given that these are of similar groups as the user has expired before. Additionally, the users themselves can search for new different types of content causing the system to also include these in the recommendations.

4.1.2. Stakeholders

Users: Utilising content-based filtering for the recommendation system would mean that the system would only rely on a user's personal history instead of other users. For how normal users engage with the available content, nothing changes apart from the type of content they get recommended. These recommendations will be more specific to a user's interest regardless of what other similar users will like. As for users that create content themselves this means that they would create more general content to cater to more users rather than only creating a niche genre of content.

Platform owners: For Instagram's platform owners having a content-based filtering recommendation system in place means that having the content posted being grouped (through for example hashtags) should be done on a broader rather than narrow basis or assigning multiple groups to content. This as the content would otherwise never get recommended to its users.

Advertisers: Similarly to users that are content creators, advertisers would also attempt to make their advertisement broader rather than targeted to get recommended more often.

Privacy regulators: No guidelines need to be established on the basis of recommendations from 'similar' users such as the spread of misinformation as that is not used in this type of recommendation system. However, there do need to be guidelines established based on the user's own modelled profile and the content that may be incorrectly grouped. This can be done for example through means of transparency.

4.1.3. Trade-offs

The system provides user independence as content is recommended based on a user's exclusive rating without the use of other users' profiles. This makes the system very easy to scale. This process is solely done by the system itself so no user feedback mechanism is incorporated by the system. This means that the system cannot ensure the recommendations made are correct. This happens when a user might interact with content that he or she is not interested in, whereas the system views this as a positive interaction. Thus resulting in an unfavourable recommendation.

Another trade-off is that the system provides transparency based on why certain content is recommended by looking at what other similar content profiles are highly rated in the user's profile. This makes the recommendations fine-tuned to a user's niche interests. The downside to this is that only similar content is recommended to what the user has interacted with before, so no new and different content will be recommended.

A last trade-off is that new content that is added is readily available to be recommended as no consensus is needed with the system only looking at the similarity of the items. However, the system might struggle with automatically generating features for certain kinds of content. Also, the system suffers from the cold start problem in the case of new users who have not interacted with any or not much content yet and can not easily make recommendations to the users who have been inactive for long periods.

4.2. Collaborative-filtering

Collaborative filtering is a process of filtering posts through the opinions of other people [14]. That is, the models recommend a post to a user X based on the interest of a similar user Y. This is visually shown in Figure 4.2 below. While collaborative filtering is relatively new in the computer science community, humans have been doing it for centuries. It may be buying an item, watching a movie, or trying out new restaurants based on human opinions with similar tastes. This recommendation technique can be applied to Instagram's recommendation system by suggesting users' posts/reels based on the behaviour of other similar users. This can be achieved using collected user data such as likes, comments and shares to build a user-post matrix. A user-post matrix consists of a table where each row is represented as a user and each column is represented as a post as shown in Table 4.2 below. The absence of value in a cell indicates that the user has not yet interacted with the post.

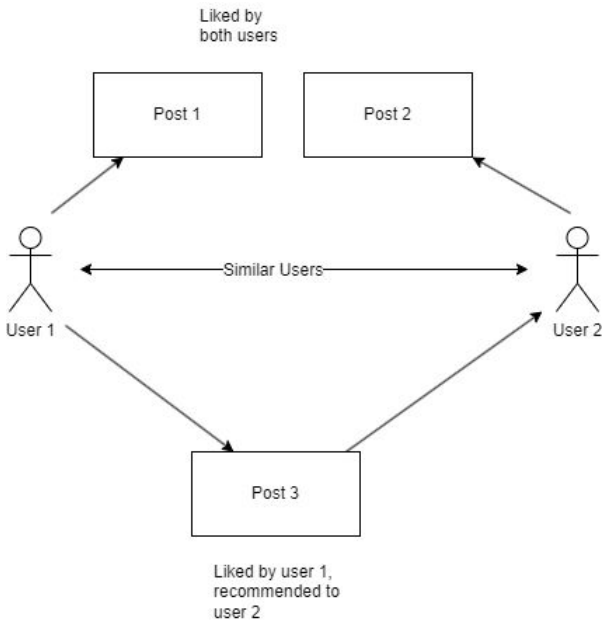

Figure 4.2: Collaborative filtering

In this case, the ratings are scalar, which consists of numerical ratings. For example, commenting on a post shows more engagement than liking a post. The data gathered for the user-post matrix is achieved through implicit means. That is, the data gathered are inferred from the user's actions. For example, if a user likes, comments or shares a post or rewatches it multiple times shows that the user is interested in the specific post.

	#cat	*#dog*	*#elephant*
Mike	Like	Share	Comment
Florence	-	Like	Comment
Adam	Comment	Like	Like
Emilia	Like	Like	Like

Table 4.2: Example user-post matrix for collaborative filtering

4.2.1. Quality attributes of the system

Scalability: One of the key quality attributes of collaborative-based systems is that they are scalable [12, 6]. These systems are designed to work on very large data sets. In Instagram's case, this is crucial. Instagram has over a billion users and it's important to deliver content to all these users and others who are constantly signing up for the platform.

Performance: The performance of collaborative-based systems is usually good for a large number of users [15]. Amazon has been using item-based collaborative filtering since 1998 and has been a successful recommendation system for them. Amazon has millions of users to which the recommendation system is applied without any performance issues. Instagram has over a billion users and it's important to deliver content to all these users without any latency. In terms of performance, recommending content to users using collaborative-based systems will be an advantage.

Flexibility: Collaborative systems are flexible and adaptive. As the user's preferences change over time, the recommendations given using collaborative filtering take that into account. This makes sure that the user is recommended the most up-to-date content according to their preferences. This is a key quality since people's preferences change over time e.g. due to age or other factors, it's necessary to keep them engaged using personalization over time and thus driving user retention.

4.2.2. Stakeholders

Instagram's recommendation system consists of many stakeholders. These include Instagram users, advertisers, platform owners, content creators and regulators, and privacy advocates.

Users: The most important stakeholders for the Instagram recommendation system are its users. They share their content such as photos, stories, reels, etc., and follow other users to engage in their content. The goal of the recommendation system is to provide users with personalized content that will increase their engagement with the platform. Collaborative filtering as the recommendation system would mean that the personalized recommended post will be highly influenced by the activity of similar users.

Advertisers: The advertisers that use Instagram as a marketing platform rely on their recommendation system to target their ads to the correct audience. This means that their ads will be directed towards the same type of audience when collaborative filtering is used as the recommendation system. For example, if there is a pool of five similar users and three of them engage with football content, the other two will likely be recommended for football content as well. Next, the platform owners of Instagram are Facebook and are also considered stakeholders of the recommendation system. Their primary goal is to increase user engagement and retention, which is influenced by the recommendation system. The better the recommendation system performs, the higher the user engagement and retention and thus the higher the revenue.

Content Creators: The better the recommendation system performs, in this case, collaborative filtering, the larger audience they will be able to reach and gain more followers.

Privacy Advocates: They are concerned about privacy issues of the users and may fear the spread of misinformation which directly ties in with the recommendation system.

4.2.3. Trade-offs

One of the main advantages of the collaborative filtering technique is personalization. It provides personalized recommendations to the user based on their preferences and past interactions with the system [9]. This leads to a better user experience overall. Lastly, collaborative-based systems are scalable [12]. They can handle large data at once. This is a make-or-break point for Instagram as it has more than a billion users and thus requires a scaleable recommendation system. Lastly, collaborative systems are adaptive. As the users' preferences change over time, so do the recommendations. One of the main disadvantages of collaborative-based systems is the first-rater problem [10]. They entirely rely on users' preferences to make recommendations to other users. Until a post has not been engaged enough, the system would not be able to recommend it to other users. This can make it difficult to provide recommendations for new users or posts. The second disadvantage of collaborative systems is transparency [10]. Collaborative systems are black boxes and the only explanation a post is recommended to a user is that an unknown pool of users have a similar taste and have liked, commented or shared the post.

4.3. Context-aware recommendation system

There are contextual factors that may cause user preferences to change over a period of time [14]. Most conventional approaches that were mentioned before, do not incorporate this. Therefore Context-aware Recommendation systems were developed. Here the context is multidimensional in nature and can be categorised into two types, namely, 'static' where its attributes and structure remain the same and 'dynamic' where those change over time (e.g attributes becoming obsolete or new relevant situations being added to the system). Additionally, the system does not necessarily fully observe the contextual attributes and its structure but can also only partially observe certain aspects of it or not know anything explicit at all. This means that the system has to learn user preferences in various contextual situations to model the multidimensional user-content space accurately. This modeling can be done through the usage of computational intelligence techniques such as that bio-inspired computing techniques (e.g. Artificial Neural Network, Genetic Algorithm, etc.) and also statistical computing techniques (e.g. Matrix Factorization, Hidden Markov Model, etc.). The purpose of using bio-inspired techniques is to generally handle complex problems such as handling multidimensional data. For example in optimizing the recommendation list. On the other hand, statistical computing techniques are more used for the system to learn and understand the nature and relationships among data. Such is the case when the system tries to latent context knowledge.

4.3.1. Quality attributes of the system

Performance: Due to the incorporation of contextual attributes for modeling a user's profile, the system can make better recommendations that fit a user's interests under different contexts. In other words, more accurate results are provided in the recommendation that the user most likely would interact with.

Flexibility: With the way the system categorizes the various types of contexts, it can adjust itself to match a user's interests even if it changes over time which means that the recommendations based on those will also be adjusted.

4.3.2. Stakeholders

Users: Having a context-aware recommendation system in place compared to more conventional approaches would make the recommendations an even more appealing feature for its users. This as it caters to their changing interests over time and does not only recommend them narrow types of content but also on a broader scale. However, it should be noted that users can still be wary of the collection of implicit contextual attributes which they may not necessarily agree with.

Platform owners: The additional usage of contextual features for the recommendations means that the platform owners should take deliberate care and responsibility for how Instagram as a platform handles this.

Advertisers: Advertisements can be made that can utilize this contextual information to be more targeted to its users. This is to increase their chances of potential customers buying their advertised

product.

Privacy regulators: Strict guidelines need to be established on various aspects of the collection of contextual information of Instagram's users. This is in the way of how it is collected, how it is stored, how well the users are informed, and also how the latter can manage their own collected information.

4.3.3. Trade-offs

The context-aware recommendation system, just like other types of systems, also suffers from common problems such as cold start, data sparsity, and scalability [14]. Cold start forms a problem as the system heavily relies on user and content information from which there is a lack of information at the beginning. With static and observable context, the system can directly learn contextual information. On the other hand with dynamic and (partial) unobservable context the system has to use context inference or explore through continuous feedback under different context situations.

Data sparsity also forms a problem due to the multidimensional nature of context causing available user preferences to be narrowed down. To mitigate missing user preferences, prediction is done on what is observed or through various techniques such as context similarity, latent knowledge, etc.

Scalability is also an issue due to now not only dealing with user and content information but also including context where the system has to work in multi-dimensional space. Here using more contextual information leads to an increase in dimensions the system has to work with. To improve this process, various dimensionality reduction techniques and/or context weighting approaches (for dynamic contexts) can be used.

Another issue the context-aware recommendation system has to face due to its inclusion of context is the privacy of its users. Collection of the necessary contextual attributes through implicit feedback from the user and the usage to make recommendations requires proper privacy considerations. With how these are set up, it can not only influence how users would interact with the system but also affect the performance of the system. For example, limiting the amount of contextual information that the system can gather, can lead to the data sparsity problem and result in lower accuracy of its recommendations.

4.4. Hybrid recommendation system

Each of the recommendation systems has its strengths and weaknesses. All the learning-based recommendation systems techniques such as content-based and collaborative filtering suffer from the cold start problem in different forms [16]. The recommendation systems are usually combined for better performance and are known as hybrid recommendation systems. Combining the recommendation systems helps to avoid the limitations of content-based and collaborative systems [2]. There are many ways to combine recommendations such as: [1]

1. Implement collaborative and content-based methods separately and combine their predictions.
2. Implement a collaborative approach with content-based characteristics.
3. Implement a content-based with collaborative characteristics.
4. Implement a general unified model that incorporates both collaborative and content-based characteristics.

There are different ways the recommendation systems can be combined to form hybrid recommendation systems which include:

Weighted: The scores of the different recommendation systems are combined to get one single recommendation. For example, it may be a linear combination of the two recommendations that were given by each recommendation system. The weights can later be adjusted according to user feedback.

Switching: The recommendation system switches between the recommendation techniques based on the current situation. For example, the content-based recommendation system is used first. If it

can't recommend a post with a good confidence score, then the collaborative recommendation system is used. [3]

Mixed: The recommendations from different systems are given at the same time.

We will use the weighted method to combine different hybrid recommendation systems as shown in Figure 4.3 below. The user will be able to change the weights for each of the systems according to their own preferences. This will not only allow the user to make their own choices but will increase transparency in the recommendation system architecture.

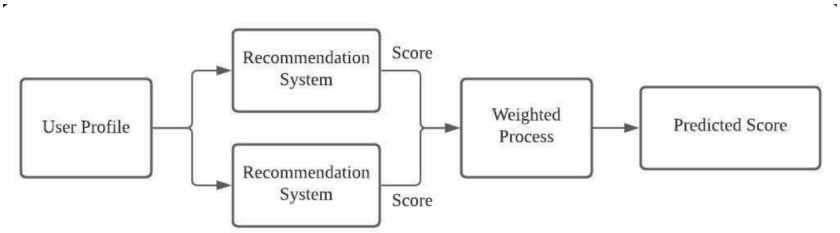

Figure 4.3: Combining recommendation system using the weighted technique

4.4.1. Quality attributes of the system

Improved Accuracy: The results show that the hybrid recommendation systems developed in MESH provided higher accuracy than the state-of-the-art collaborative filtering and content-based recommendation systems [12].

Diverse Recommendations: The hybrid recommendation systems allow for diverse recommendations. They balance between exploring and exploiting by using two recommendations from two different systems and choosing one of them. This allows the user to explore different recommendations while still taking the user profile into account.

Flexible: The hybrid recommendation systems can be designed to be flexible that incorporate new techniques or changes to the weighting of different techniques over time. This allows for adapting the recommendation system according to user preferences.

4.4.2. Stakeholders

Users: The primary stakeholders of the system are the users who will receive the recommendations. Users would want accurate and personalized, yet diverse recommendations based on their behavior and interests.

Advertisers: They want to ensure their ads are being shown to the right users, but also to users who are ready to explore new things and will engage with the ad post.

Content Creators: They want to ensure their content is shown to the right users, but also to users who are ready to explore new things and will engage with their content.

Regulators and Policy Makers: They want to ensure the system is unbiased, transparent, and doesn't violate user privacy or ethics.

4.4.3. Trade-offs

The hybrid recommendation systems can mitigate a lot of problems collaborative filtering and content-based recommendation systems face, however, it doesn't completely fix the ramp-up problem since it depends on the results of content-based recommendation systems that do have the problem [4]. They

provide accurate recommendations when trained with large data sets that guarantee reliable evidence of user interests. They provide more diverse recommendations that incorporate different techniques, always recommending different but interesting items for the user. Some of the disadvantages include complexity, computational resources required, and transparency of the system.

5 Proof-of-Concept

To demonstrate and test the effectiveness of alternative recommendation systems for Instagram, a proof-of-concept (PoC) with an experiment was developed. This section will go into depth on details such as the overall purpose of the prototype, its implementation, and its evaluation of it.

5.1. Purpose of the PoC

Aside from the theoretical perspective, it is also important to consider the practical perspective of alternative recommendation systems for Instagram using a PoC. The purpose is to test to identify potential issues and limitations, gather feedback to improve the system and in the end demonstrate the effectiveness in generating personalized content recommendations. During the development process, it is crucial to carefully select components that should be implemented or not, as only those that influence the recommendations generated should be implemented. This approach helps limit the PoC's scope and ensure that only necessary functionalities are implemented, making it easier to complete the PoC within the given timeframe.

Our focus with the PoC was creating a system that generates recommendations based on a user's interactions with a post, such as liking, commenting, and sharing. With this in mind, Instagram's post-related functionality was implemented, whereas other functionalities such as authentication, direct messaging, and sharing content were deemed unnecessary and therefore left out.

5.2. Implementation of the PoC

Similarly to what Instagram is built on, the Python-based web framework Django[5] is used as the technology stack for proof-of-concept implementation. The Django framework adheres to the Model-View-Controller pattern and allows for quick and efficient development as its setup provides a basic structure of the application, including settings, URL routing, and middleware.

5.2.1. Front-end

Django provides basic functionality for front-end development, including templates, static files, forms, and URL routing. Django offers a templating engine for the easy creation of HTML templates with dynamic content placeholders. Additionally, Django also has static file handling functionality to use files like CSS, JavaScript and images and form handling functionality for easy creation and processing of HTML forms. Lastly, to make it easy to organize and maintain the application, URL patterns are used to map the URLs defined in the configuration file to the back-end views.

A combination with other front-end frameworks, like React, Angular, or Vue.js, is possible for building more complex and interactive user interfaces. However, for our use case a basic user interface compromising of a web page displaying posts from followed users, that can be liked, commented on, and shared, sufficed. Other decorative elements that mimic an Instagram page are also present in the im-

plementation but aren't interactive. Overall, our focus was on demonstrating the effectiveness of our recommendation system, rather than building a complex user interface.

5.2.2. Back-end

Django is mainly used for back-end purposes such as processing requests from the client, interacting with the database, and returning a response to the client. Under the hood, several components are used to handle the different functionalities. These are Object-Relational Mapping (ORM) for database interaction, View processing for handling user requests and HTTP responses, and middleware for modifying requests and responses. These components work together to handle requests, manage data, and generate responses to the client.

For the proof-of-concept, the core interaction functionality with the posts has been implemented in the views. Whenever a user interacts with a post through the front end, a request is sent to the back end to handle the interaction. The back-end then checks the database for existing data and updates it accordingly, for example, a 'Likes' object associated with a post and the user will either be removed with the 'Post' field 'likes' decreased or a new one will be created with the 'likes' increased. After any interaction is handled in the back end, an HTTP response is sent to the front end, refreshing the page to reflect the updated changes. The alternative recommendation system has been implemented in the back-end, which will be further elaborated upon in subsection 5.2.4.

5.2.3. Database

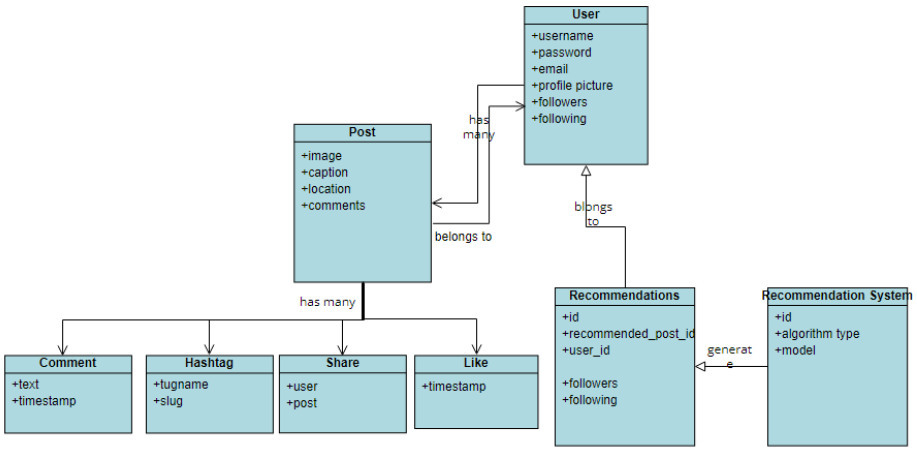

Figure 5.1: Database UML scheme

The Django framework comes with SQLite, which is a lightweight, cross-platform, easy-to-use, fast, and efficient database system that is fully ACID-compliant and open source. Due to its nature as an embedded database, it allows for easy deployment and management within a code base without requiring a separate server to run on. Through Django models, the structure of the SQLite database tables can be specified and CRUD (Create, Read, Update, Delete) operations can be performed. The proof-of-concept database scheme can be seen in Figure 5.1, which does not specify the exact implementation details but only encapsulates the main components and their relationships. A migrations framework is included in Django's ORM that allows for database scheme changes over time, while maintaining backward compatibility with previous versions of the schemes. Additionally, migrations automatically generate SQL statements, making the process and maintainability of the database easier. Django also offers a built-in administrative interface that can be used for easy management of database records and also testing and debugging of the application during development.

5.2.4. Proposed recommendation system

For the proof-of-concept, we implemented content-based, collaborative filtering and hybrid recommendation systems, which can address the drawbacks of the current recommendation system of Instagram. The drawbacks of the current recommendation system include echo chambers and homogenization of the content that is analyzed in the previous section. Moreover, the current recommendation system is a black box and lacks transparency. In the process of constructing the proof of concept, we followed these steps:

a. Dataset preprocessing: We used a publicly available dataset that contains user interactions with the post, the number of upvotes of the post, as well as the content of the posts. For this dataset, we remove any duplicated, missing data and merge CSV files to ensure data consistency. We also normalize the data to make sure it can be used for training and testing.

b. Implementing a content-based recommendation system: We implemented a content-based filtering recommendation system that uses a pre-trained model to calculate similarity to measure the similarity between items based on their extracted features. The system then recommends items with the highest 10 similarity scores to a user's previously liked content.

c. Implementing collaborative recommendation system: We implemented an item-based collaborative recommendation system. For item-based filtering, we used cosine similarity to measure the similarity between items and predicted user preferences based on users' interactions with similar items.

d. Implementing a hybrid recommendation system: We implemented the hybrid recommendation system which combines collaborative filtering and content-based filtering by using weighted techniques. The system calculates the similarity scores with both methods and sums them up.

5.3. Evaluation of the PoC

To evaluate the performance of each alternative recommendation system, we conduct an experiment in which we test each recommendation system architecture on a publicly available posts' dataset [1] and use MRR metrics to evaluate the result.

5.3.1. Testing Data

The dataset we use for the experiment on the recommendation system contains 3 CSV files which are post data, user data, and view data. The post data contains a title that briefly summarizes the content of each post and its category and post ID. The user dataset contains users' personal information such as Id, name, and gender. At last, the view dataset includes the likes count and likers count of each post.

5.3.2. Evaluation Metrics

MRR [11] is an evaluation metric used in information retrieval and recommendation systems. It measures the average rank of the first relevant item in the list of recommended items. Testing recommendation algorithms in a PoC is usually done on a small group of users to collect feedback to evaluate and refine the system. We did not do that as it's difficult for the given timeframe of this project and we only did the MRR evaluation.

$$\text{MRR} = \frac{1}{Q}\sum_{i=1}^{Q}\frac{1}{\text{rank}_\text{i}} \tag{5.1}$$

In this formula, Q is the total number of recommendation results for each query, and $rank_i$ is the rank of the first relevant result for query i. The reciprocal of $rank_i$ is used because the lower the rank of the first relevant result, the higher the MRR score.

[1] https://www.kaggle.com/datasets/vatsalparsaniya/post-pecommendation?select=user_data.csv

5.3.3. Results

For each alternative recommendation system in the PoC, the input is a brief description of a certain post, and the recommendation system will give a set of the best 10 posts as its recommendations. We select 5 queries for our experiment and compare the MRR scores of results generated by each recommendation system. As shown below, the hybrid recommendation systems outperform other systems.

Table 5.1: Comparison of MRR scores of different recommendation systems

Recommendation System	Query 1	Query 2	Query 3	Query 4	Query 5
Content-based	0.251	0.100	0.100	0.192	**0.164**
Collaborative	0.102	0.000	0.049	0.06	0.037
Hybrid	**0.279**	**0.150**	**0.183**	**0.195**	0.162

5.3.4. Analysis:

As shown in the above table, the performance of the hybrid system is the best of the three alternative recommendation systems in most cases, which is close to what we expect. This is because, in the hybrid recommendation system, the strengths of each method can be combined while their limitations can be mitigated. The content-based recommendation system may perform well in suggesting posts that are similar to what a user has already interacted with, while the collaborative filtering method may be better at capturing the diversity of items preferred by other users.

6

Quality assurance

An essential part of any software development process is Quality assurance. It is used to help validate the implemented functionality by ensuring the requirements and specifications are met and help identify potential bugs or errors in the codebase. For the development of the proof-of-concept implementation, a variety of best practices have been followed such as the use of a Testing suite, Static Code Analysis Tools, and a Continuous Integration pipeline.

6.1. Code Quality

Ensuring code quality requires us to make sure our code is consistent, readable, and maintainable. For that, we followed best coding practices that involve writing readable and maintainable code. This is utilizing variable name conventions, documentation in ways of effective comments and docstring (pydocs) to enhance the user's understanding of the code, organizing the code into logical functions and classes, and refactoring and optimize the code where necessary to avoid duplication and unnecessary operations. In addition, Static analysis tools, such as *pylint, bandit, mypy, and flake8*, help check code formatting and detect errors. It's also essential to test and debug the code thoroughly to ensure it works correctly and is free from errors, which will be explained more in-depth in section 6.2. Lastly using Git and GitLab tools, such as branches, issues, and a CI/CD pipeline, helps manage the code effectively and facilitated collaboration among the team members. By following these guidelines and using the right tools, we made sure our code is written in an efficient, effective, and readable manner.

6.2. Software quality testing

Django offers software testing out of the box to ensure developed applications are of quality and without unintended problems such as bugs and errors. For the proof-of-concept implementation, Django's in-built robust testing framework has been utilized to create tests for different components such as URLs, views, models, and forms:

- Testing URLs is done by checking the URL handling for a particular view function. Here Django provides the 'reverse()' function which maps a URL using defined patterns to a view function and the 'resolve()' function does the opposite by mapping a view name to a URL and filling in any parameters in the URL pattern.
- For testing the Views, Django provides the 'Client' class to simulate requests to a view and assert the responses being correct.
- Testing the Models, Django provides the 'TestCase' class which allows interaction with the database. Tests can be used to check a model instance creation and saved it to the database.
- To test Forms, the 'TestCase' class can be used again to test the behavior of form fields, validation of the form, and submission of the form contents.

6.3. Continuous Integration

A Continuous Integration pipeline is used to automate the process of building and testing the implementation to ensure the code is up to the required software quality standards. This helps in discovering and more easily fixing otherwise unnoticed bugs and other issues during the development cycle. For each commit, the pipeline will be run and create a Python environment and install the dependencies in the *'requirements.txt'* file. Afterwards, it executes the following stages in consecutive order:

- **Build**: A database migration is performed to ensure the latest database scheme is used and minimizes the problems caused by outdated models. Afterwards django check command is run to inspect that the project has no problems.
- **Checkstyle**: The different Static Analysis tools will be run to ensure the code is properly formatted and to ensure no potential bugs are in the code.
- **Test**: The Test suite will be run to ensure no changes are introduced in the code that might break existing functionality. To see how well the tests overall cover the codebase, code coverage is measured and displayed in the pipeline terminal.

7

Further Improvements

For future work, a suggested improvement includes researching different types of recommendation systems through a user study. Our current evaluation is only an empirical analysis using the MRR metric and does not utilize the developed Proof-of-Concept as much. Additionally performing a user study helps to gain valuable insights into usability issues, validate our assumptions more, gather feedback on potential issues and limitations, and lastly help refine the users' needs regarding the recommendation system. Besides this, another suggestion involves performing experiments using different datasets, as currently we only used a dataset gathered from a single source. Using more datasets that are varied can improve the used models by making their recommendations more general and robust which improves their accuracy as the recommendations are more representative of the general user's needs. A last suggestion involves improving the implemented recommendation system to make more use of a user's activity. Currently, only likes, comments, and shares are used but other activities such as following users, uploading posts and sharing stories can also be taken into account for the recommendations.

References

[1] Gediminas Adomavicius and Alexander Tuzhilin. “Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions”. In: *IEEE Transactions on Knowledge and Data Engineering* 17.6 (2005), pp. 734–749. DOI: 10.1109/TKDE.2005.99.

[2] Marko Balabanovic and Yoav Shoham. “Fab: Content-Based, Collaborative Recommendation”. In: *Communications of the ACM* 40 (Mar. 1997), pp. 66–72. DOI: 10.1145/245108.245124.

[3] Marko Balabanovi. “Exploring Versus Exploiting when Learning User Models for Text Recommendation”. In: *User Modeling and User-Adapted Interaction* 8.1 (Mar. 1998), pp. 71–102. ISSN: 1573-1391. DOI: 10.1023/A:1008205606173.

[4] Robin Burke. “Hybrid Recommender Systems: Survey and Experiments”. In: *User Modeling and User-Adapted Interaction* 12 (Nov. 2002). DOI: 10.1023/A:1021240730564.

[5] Django Software Foundation. *Django*. Version 2.2. May 5, 2019. URL: https://djangoproject.com.

[6] Thomas George and Srujana Merugu. “A scalable collaborative filtering framework based on co-clustering”. In: *Fifth IEEE International Conference on Data Mining (ICDM’05)*. 2005, 4 pp.-. DOI: 10.1109/ICDM.2005.14.

[7] *Instagram’s algorithm officially listed as the cause of death in a court case in the UK- Technology News, Firstpost*. [Online; accessed 9. Apr. 2023]. Oct. 2022. URL: https://www.firstpost.com/tech/news-analysis/instagram-listed-as-official-cause-of-death-in-a-14-year-old-girls-suicide-case-in-uk-11378201.html.

[8] Aman Kharwal. *Instagram recommendation system with machine learning: Aman Kharwal*. June 2022. URL: https://thecleverprogrammer.com/2022/06/14/instagram-recommendation-system-with-machine-learning/.

[9] Saurabh Kulkarni and Sunil F. Rodd. “Context Aware Recommendation Systems: A review of the state of the art techniques”. In: *Computer Science Review* 37 (2020), p. 100255. ISSN: 1574-0137. DOI: 10.1016/j.cosrev.2020.100255. URL: https://www.sciencedirect.com/science/article/pii/S1574013719301406.

[10] Neal Lathia et al. “Temporal diversity in recommender systems”. In: *Proceedings of the fourth ACM conference on Recommender systems*. July 2010, pp. 210–217. DOI: 10.1145/1835449.1835486.

[11] Dragomir R. Radev et al. “Evaluating Web-based Question Answering Systems”. In: *Proceedings of the Third International Conference on Language Resources and Evaluation (LREC’02)*. Las Palmas, Canary Islands - Spain: European Language Resources Association (ELRA), May 2002. URL: http://www.lrec-conf.org/proceedings/lrec2002/pdf/301.pdf.

[12] Francesco Ricci et al. *Recommender systems handbook*. New York; London: Springer, 2011. DOI: 10.1007/978-0-387-85820-3.

[13] Sunita Sarawagi. *Instagram architecture and database – how does it store and search billions of images*. Accessed on March 17, 2023. Dec. 2022. URL: https://scaleyourapp.com/instagram-architecture-how-does-it-store-search-billions-of-images/.

[14] J. Ben Schafer et al. “Collaborative Filtering Recommender Systems”. In: *The Adaptive Web*. Springer, Jan. 2007, pp. 291–324. DOI: 10.1007/978-3-540-72079-9_9.

[15] Gábor Takács et al. “Scalable Collaborative Filtering Approaches for Large Recommender Systems”. In: *Journal of Machine Learning Research* 10 (2009), pp. 623–656.

[16] Poonam B Thorat, R Goudar, and Sunita Barve. “Survey on Collaborative Filtering, Content-based Filtering and Hybrid Recommendation System”. In: *International Journal of Computer Applications* 110.1 (2015), pp. 31–36. DOI: 10.5120/19308-0760.

[17] Nailong Zhang. *Improving Instagram notification management with machine learning and causal inference*. Oct. 2022. URL: https://engineering.fb.com/2022/10/31/ml-applications/instagram-notification-management-machine-learning/.

A

Team rubrics

A.1. Team background

Irtaza Hashmi I graduated with a bachelor from TU Delft. I have around 1.5 - 2 years of experience in software projects, where mostly I worked on Data Science and Machine Learning projects. For this course, I would like to learn how to build, maintain, and scale software using architectural techniques such that the code is concise and understandable.

Zenan Guan I gained my bachelor's degree in TU Delft. I have experience with software projects including several machine learning projects in Python, projects in website design in JavaScript, and projects in Java software development. I would like to learn the knowledge about how to analyze the architecture of an existing software system and how to build a software system in a professional way.

Xinrui Xu I obtained my bachelor's degree in computer science at NJUST. I have basic knowledge of Java and skills in object-oriented programming. I have some software development experiences which implemented J2EE. After taking the course, I would like to learn about how to build software architecture with high quality.

Kevin Nanhekhan I did my Bachelor's at TU Delft where I gained some experience with the front-end and back-of creating applications, as well as gaining familiarity with architectural design patterns and creating UML diagrams. For this project, I would like to improve my knowledge in the aforementioned skills by working on an existing real-world system instead of applications created from scratch which I have done in the past. In other words, as mentioned in one of the lectures, putting the focus more on building the right system instead of building the system right.

A.2. Team objectives

Our choice of an existing system is that of Instagram. This is because Instagram is one of the biggest social media platforms where its users can share photos and videos through its services with their followers. The stakeholders include ordinary users, influencers, advertisement providers, and investors. The interesting part to look at would be the recommendation system which suggests posts based on one's activities, likes, and comments. Another interesting aspect to look at is its way of ensuring the availability and scalability of its services through means of using a distributed system and its database design (master-slave replication and partitioning).

Aside from implementing our chosen system in a proof-of-concept manner, the knowledge we as a team want to gain here are:

- How to analyze an existing software system to see what exactly the architecture for it should entail. This is in terms of what the different components are and the key features of such a system. Here the focus is also on improving our skills in capturing the stakeholders' concerns for such a system. This is by formulating the different functional and non-functional requirements.

- How to represent the architecture through documentation. For example by means of UML diagrams to describe the system's architecture. Here we want to learn how we should document the specific parts of such a system in a clear way. Additionally the exploration of alternative architectural decisions
- How to evaluate the architecture by some certain standards, for example, observability, measurability, repeatability, and some other quality attributes of the architecture.

A.3. Intial Team Approach

A.3.1. Graphical User Interface (GUI)

Instagram contains a series of interfaces which allows users to interact with the software. We can build a proof of concept by creating a project which allows users to execute interactions, such as liking posts, leaving comments, etc.

A.3.2. Backend

The backend is responsible for the underlying business logic, communication with the database and user security and authentication.

Business logic

The backend will handle the underlying business logic in the application. Instagram uses a recommendation algorithm to suggest to users some posts they are interested in. Here we can build a proof of concept for this by for example recommending certain content according to which hashtags are used by a user or who a user follows.

Communication with the database

Communication with the database is also the responsibility of the backend. Any data that should be inserted, modified, or deleted will happen via the backend service.

Security and authentication

For users to interact with the application, they will need to authenticate themselves using a username and password. Users that are not signed up will have to create a new account. User security and authentication will also be implemented in the backend.

A.3.3. Database

Instagram uses a variety of databases to store data of users, including profiles of users' accounts, comments, videos and pictures which are posted by users, etc. The primary database used by Instagram is PostgreSQL. We can build a proof of concept by building a system which allows users to create accounts, leave their comments and upload media. We can build a proof of concept by building a system which contains a database to store user information securely and efficiently. The proof of concept should also include a login system with password storage and access control to ensure security. To allow for experiments/measurements, the PoC system should be able to track user behaviour and activities of the database.

A.4. Success indicators

After the completion of the course, we are planning to follow these success indicators:

A.4.1. Architecture development

- To have a better understanding of software architecture concepts by understanding the fundamental principles, patterns, and practices of software architecture.
- The ability to design, communicate and develop complex software architecture systems.
- Increased proficiency in software development by developing more efficient, scalable, and maintainable software systems.
- The ability to evaluate software architectures and make informed decisions about the suitability of an architecture.

A.4.2. Essay

- A well-written essay with a clear purpose and reasoning.
- Sound use of the theory of software architecture from academic literature for reasoning and argumentation.
- Decomposing software architecture principles for reasoning and argumentation.
- Implementing software architecture principles in practice in the group's chosen system, Instagram in this case.
- An essay written from our own perspective.

A.4.3. Proof of concept

- Clearly arguing with reference to academic literature why our proof of concept will take architectural designs.
- The proof of concept will allow us to take quantitative measurements.
- Experiments results are visualized using appropriate diagrams.
- Sound use of software engineering practices such as git, documentation, testability, etc.

A.4.4. Key Performance Indicators

- A concise essay on the software architecture of our system using academic literature with our own point of view.
- A codebase that contains the project code with all the necessary features.
- Well-documented code and instructions to compile it.

B

Accountability

B.1. Initial Agreements

At the start of the project, we all agreed to divide the work among ourselves equally to have a balance in the work everyone has to do. This is in the way of not only performing certain tasks individually but also by means of splitting our group into smaller subgroups to accomplish these tasks efficiently. In order to keep each other updated on the progress that has been made, we also agreed to meet at least once a week. Here we discuss what has been done, what still needs to be done, and a further division of tasks.

B.2. Weekly Progress

B.2.1. Week 1

Everyone: During the first week, we were looking for a team. Once we formed a team, we were choosing the system to analyze. We narrowed it down between TikTok and Instagram. Finally, Instagram was chosen as the system we wanted to choose. We spent around 14 hours per person on this course in the first week, which included attending lectures, attending planning meetings, finding groups and setting up the project.

B.2.2. Week 2

Everyone: Formulated the team plan and decided that we want to analyze the recommendation system architecture specifically of Instagram. Organized TA meetings for certain clarifications about the course and midterm requirements.

Irtaza: I worked on the team plan. Specifically, I worked on section 3 (approach) and section 4 (success indicators). In section 3, I talked about the approach we will be taking to analyze the architecture, however, we pivoted from Instagram architecture to Instagram recommendation system architecture and thus the section doesn't apply anymore. Moreover, I wrote about the success indicators that we can use to determine our progress over time. I spent around 14 hours this week which included attending lectures, attending TA and group meetings, working on the essay, and starting to research the topic.

Zenan: For the team plan, I worked on section 2 (team objectives) and part of section 4 (key performance indicators). This week, I spent around 14 hours attending lectures and meetings, researching the topic, and writing the essay.

Xinrui: We wrote our team plan during week 2 and I was responsible for the part of the backend and proof of concepts in section 3. This week, I started to learn about Instagram and its background information. I spent about 13 hours on lectures, meetings, and research.

Kevin: During the writing of the team plan, I was responsible for writing parts of section 2 (team objectives) as well as general writing additional information to the other sections. The writing took around 6

hours which also included discussion with the team on what to write and division of who writes what.

B.2.3. Week 3

Everyone: Receiving feedback on our Team plan as well as getting more details on what is expected for the project.

Irtaza: Started to work on the midterm essay. First, I started working on the essay structure in detail. Contributed to the team by discussing what should be included in the sections and subsections of the essay structure. Next, I researched alternative recommendation systems by reading literature and articles. Mainly, I was responsible for writing about collaborative filtering and hybrid recommendation systems. I spent around 14 hours this week which included attending lectures, attending TA and group meetings, and completing the tasks above.

Zenan: Did research on the software structure of Instagram and revised the team plan with teammates, and also start to work on the midterm report. This week I spent around 14 hours attending the lectures, discussing, and researching.

Xinrui: I did some research for the midterm essay, including the current structure of Instagram. And I revised a team plan with my teammates. For this week, I spent nearly 14 hours on group meetings, lectures, research, and writing a midterm report.

Kevin: Helped out with writing the essay structure following Minto's pyramid principle. I also researched some alternatives to the recommendation systems and was responsible for writing the sections on content-based filtering as well as context-aware recommendation systems. This in total took around 14 hours which also included meeting with the team to discuss the structure and divide the writing sections.

B.2.4. Week 4

Irtaza: Researched alternative recommendation systems by reading literature and articles. Mainly, I was responsible for writing about collaborative filtering and hybrid recommendation systems. Moreover, worked on a proof-of-concept of the architecture by making interfaces in GitLab. I spent around 14 hours this week which included attending lectures, attending TA and group meetings, working on the essay and setting up the PoC.

Zenan: Worked on the midterm essay and was responsible for part 1 (Instagram - Product Vision and Problem Analysis) and part 2 (Main Architectural Style). I spent around 14 hours attending the lectures, researching the architecture of Instagram, and writing.

Xinrui: Started to work on the midterm essay and I was responsible for the analysis of the current recommendation system (section 2) and introducing one alternative recommendation system for Instagram (section 3). I spent around 14 hours attending the lectures and group meetings, researching the current recommendation system and social network alternative recommendation system, and writing them in the report.

Kevin: Continued research on the recommendation system alternatives of content-based filtering and context-aware recommendation systems and finalized the writings for the midterm essay on those. The research took around 6 hours and writing around 4 hours. Additionally helped with working on the proof-of-concept code which took around 4 hours which includes meeting discussions on the code and working in liveshare.

B.2.5. Week 5

Irtaza: Worked on the front end of the proof-of-concept by developing the user interface (UI) of Instagram using the Python-based web framework, Django. Developed features that allowed the user to like, comment and share posts using the UI. Moreover, started to develop the initial content-based recommendation system. I spent around 16 hours this week which included attending lectures, TA and

group meetings, and mostly working on the PoC. Also, I peer-reviewed the midterm essays and PoC.

Zenan: Studied the recommendation systems, worked on the analysis of the container view of Instagram and made the diagram. It took me around 15 hours to finish my job this week

Xinrui: Worked on the analysis of the recommendation system section and part of the C4 model section. It took me around 15 hours to finish the task and attend lectures and group meetings.

Kevin: Setting up the Django framework used for the Proof-of-Concept and worked on the database models. This took around 14 hours which included time for researching the framework, setting it up and actual implementation where I also had to get familiar with it.

B.2.6. Week 6

Irtaza: Continued to work on the PoC by developing the content-based recommendation system. I spent around 14 hours this week which included attending lectures, attending TA and group meetings, and mostly working on the PoC.

Zenan: Worked on the analysis of the component view of Instagram and made the diagram, search and write key scenarios. In total it took me 13 hours.

Xinrui: Added new content to the previous section including the domain model analysis, external dependencies, and recommendation system analysis. It took me 13 hours in total.

Kevin: Continued work on the PoC and also connected the database models, back-end views, and front-end templates together. This in total took around 16 hours.

B.2.7. Week 7

Irtaza: Worked on the report and improved sections according to the peer review comments. I spent around 14 hours this week which included attending lectures, attending TA and group meetings, improving the report, and making UML diagrams.

Zenan: Worked on the PoC of the hybrid recommendation systems, processed the data and modified the diagrams of containers and components view, added descriptions. I spent 9 hours this week on data processing and 4 hours modifying the diagrams.

Xinrui: Completed the coding of alternative recommendation systems in our PoC, and tested their performance on the dataset. I spent 8 hours coding for each alternative recommendation system and testing them on the dataset, 6 hours writing the experiment of the PoC section of the report, and 2 hours attending meetings.

Kevin: For the Proof-of-Concept section in the report, wrote the Focus and the first three subsections on the Implementation. Additionally wrote The Quality Assurance section. Writing the sections took around 5 hours. Lastly, created the Test suite and setup a simple CI/CD pipeline. This in total took around 10 hours to do from researching how this can be done for a Django project to setting it up and implementing it in the PoC.

B.2.8. Week 8

Everyone: Prepared for the presentation by creating the slides, making a division, and performing several dry runs.

Irtaza: Worked on the report by working on the design problem section of the report. Moreover, started to work on the final presentation by setting up the slides and creating slides for the alternative recommendation system. I spent around 15 hours this week which included attending group meetings, working on the PoC, and the essay and preparing for the final presentation with the group.

Zenan: Worked on the slides on the main architecture style and prepared the presentation, rehearsed with teammates. Overall I spent around 14 hours preparing the presentation.

Xinrui: Worked on the slides of experiments and analysis. I spent overall 14 hours attending group meetings and preparing for the presentation.

Kevin: Worked on the slides on the PoC. This took around 10 hours which includes planning the overall presentation with the team, preparing the slides and meeting up with the team to perform some practice runs. Additionally performed Static Analysis and fixing the CI/CD pipeline for 5 hours.

B.2.9. Week 9

Everyone: Finalised the PoC and the report before submission. Reread the report and correct any type of spelling and grammar mistakes.

Irtaza: Additionally, restructured the report to the pyramid structure and compared the essay to the rubric provided in the lecture slides. I spent around 15 hours this week which included group meetings and finalizing the essay and the PoC.

Zenan: Improved the report, checked grammar, added descriptions of stakeholders as well as drew the stakeholder matrix. I spent around 14 hours this week finalizing the report

Xinrui: Refined the code of the alternative recommendation system. I also improved the experiment section in the paper. I spent around 14 hours finishing the task above.

Kevin: Improved the general writing especially of the PoC and Quality assurance sections and add the last changes to the PoC. This in total took around 14 hours which also included meeting discussions.

B.3. Reflection

The actual work done and the roles of each team member were fulfilled. We managed to submit a successful team plan, plan the essay structure, implement the PoC, present our results to our peers and finally write the final essay. Initially, what needed to be done was not fully clear to all of us but it got clearer over time. As a team, we reflected on this and the tasks that needed to be done were made clearer in the midterm peer feedback. As a result, we managed to successfully finish the final version of the report, as well as create a working Proof-of-Concept that includes the functionality we wanted to have. Looking at our defined Success indicators (see Appendix A), we all agree that these have been achieved.

C

GitLab Repository

https://gitlab.ewi.tudelft.nl/in4315/2022-2023/sa-team20