REPORT

인스타그램 어플의 객체 조사

과목명 | 객체지향프로그래밍
담당교수 | 윤성혜 교수님
학과 | 소프트웨어학부
학년 | 2학년
학번 | 20171664
이름 | 이범석

목차

1. 서론

21세기에 들어 인터넷이 급격히 발전되고 대중화되면서 사람들은 인터넷을 통해 소통하기 시작했다. 이러한 사회적 배경 속에 SNS가 생겨났고, 폭발적인 발전을 하게 되었다. 많은 SNS 플랫폼이 있지만 필자는 휴대폰 어플리케이션 중 '인스타그램'을 많이 사용한다.

'인스타그램'은 10/20대 사이에서 '페이스북'과 함께 가장 많이 사용되는 SNS 플랫폼이다. 각자 일상 속 찍은 사진과 함께 해시태그를 달아 관련 게시물을 볼 수도 있고, '스토리'를 통해 일상을 공유하여 'DM' 이라고 불려지는 메시지를 보내 소통을 할 수도 있다. 하지만 하루에도 몇 시간씩 이 어플과 기능을 사용하면서 이 어플이 어떻게 구성되었는지는 생각해 본 적이 없었다. 이번 과제를 통해서 '인스타그램'의 기능, 객체를 파악하고, 배운 점에 대해 설명하고자 한다.

2. '인스타그램' 기능 소개

인스타그램에는 많은 기능이 있지만 주요한 기능은 크게 게시글, 스토리, 해시 태그가 있다.

i. 게시물

'게시물'은 게시자, 사진, 내용, 위치 게시 날짜 등으로 이루어져 있고, 다른 사용자들에게 공유할 수 있다. 게시물에는 좋아요, 댓글을 추가할 수 있으며, 신고, 링크 복사, 공유, 게시물 알림 ON/OFF 등의 기능이 있다. 추가로 게시자와 다른 언어를 사용하는 사람이 내용을 이해할 수 있도록 번역의 기능도 추가되어 있다.

ii. 스토리

그때그때 자신의 일상을 다른 이용자들에게 공유하고 싶을 때 이용자들은 게시물보다 스토리를 올리는 것을 선택한다. 10초 내외의 짧은 시간 동안 글과 사진으로 표현할 수 있다. 스토리는 게시물과는 다르게 사진 없이 내용만으로 구성할 수도 있고, 사진을 이용자가 원하는 크기와 위치, 방향으로 조절하여 글과 함께 추가할 수 있다. 또한 스토리의 게시자에게 'DM'을 바로 보낼 수 있는 기능이 구현되어 있다. 그리고 스토리는 받는 사람을 선택할 수 있다. 모든 이용자들에게 공유할 수도 있고, 친한 친구로 지정한 사람들이나 한 사람에게만 보낼 수도 있다.

iii. 해시 태그

'해시 태그'는 같은 태그를 가진 게시물들을 묶어서 검색 가능하게 해 주는 기능이다. 태그는 '#' 기호 뒤에 내용으로 이루어져 있다. 이용자가 궁금한 태그를 검색하면 해당 태그를 가진 보고 싶은 사진들만 볼 수 있게 해준다.

이용자들의 피드백 적용, 좋아요를 기반으로 관심 있는 게시물을 파악하여 표시해주는 기능들 또한 좋은 영향이다. 현 시대 SNS 이용자들이 원하는 기능들을 담은 덕분에 인스타그램 이용자는 급격히 증가했고, 더욱 늘어날 전망이다.

3. 객체

i. 게시물 객체

게시물에는 해당 게시물의 세부적인 정보와 게시물에 표현할 수 있는 많은 기능을 담고 있었다. 게시물이 가지고 있어야 할 특징과 바람직한 인스타그램 이용을 위한 신고 등의 기능을 포함해서 이용자들 스스로가 인스타그램의 순기능을 편리하게 이용할 수 있도록 했다. 또한 전 세계에서 이용하는 인스타그램의 특성상 다른 언어가 사용된 게시물을 편하게 볼 수 있도록 게시물 번역의 기능도 포함하여 SNS의 가장 큰 기능인 소통의 기능을 원활하게 구현한 모습이다.

게시물	
설명	해당 글의 게시자가 올린 게시물의 정보를 가지고 있는 객체
필드	게시자 / 게시 날짜 / 태그된 사람 / 사진 / 내용 / 해시 태그 / 위치 / 게시물의 하트 개수 / 하트를 누른 사람 / 댓글을 작성한 사람 / 개수 / 댓글 내용
메소드	하트 누르기 / 댓글 달기 / 다이렉트 메시지 보내기 / 게시물 저장 / 게시물 신고 / 링크 복사 / 공유 / 알림 설정 / 게시물 번역 / 게시물 삭제 / 수정

ii. 프로필 객체

모든 사용자가 회원가입과 함께 갖게 되는 프로필은 각자 사용하는 아이디만으로 알아보기 힘들 수 있기에 프로필 설명을 할 수 있게 해서 '나'라는 이용자를 다른 이용자들에게 알아볼 수 있게 하였고, 각자 게시한 게시물과 스토리를 이용자가 공개한 만큼 다른 이용자들에게 직관적으로 보여주는 모습이었다. 가장 중요한 기능이라고 할 수 있는 팔로우 / 팔로워 기능을 구현하여 해당 프로필을 가진 사람의 게시물을 볼 수 있게 하고, 내 게시물을 보고 싶어하는 사람의 목록의 정보도 가지고 있다. 이는 나의 프로필을 내가 허락한 사람만 보게 하는 기능을 가지고 있다. 또한 친한 친구 목록을 지정할 수 있게 해서 해당 게시물을 보여지고 싶은 사람에게만 나타내는 편리한 기능도 있다.

프로필	
설명	이용자마다 가지고 있는 프로필의 정보를 가지고 있는 객체
필드	아이디 / 프로필 사진 / 팔로잉 / 팔로워 / 프로필 설명 / 이메일 주소 / 전화번호 / 성별 / 게시물의 개수 / 게시물 목록 / 활동 목록 / 저장한 게시물 목록 / 팔로우 목록 / 개수 / 팔로잉 목록 / 개수
메소드	프로필 편집 / 친한 친구 설정 / 저장한 게시물 확인 / 활동 목록 확인 / 게시한 스토리 확인 / 팔로잉 / 팔로잉 취소 / 비공개 전환

iii. 검색 객체

인스타그램은 세계의 이용자가 올린 게시물들을 볼 수 있다. 내가 팔로우한 사람의 게시물 이외에 다른 게시물을 볼 수 있도록 한 기능이 검색 기능이다. 계정, 해시태그, 위치를 검색할 수 있는 기능을 구현하여 이용자가 검색하고자 하는 정보와 게시물을 효율적으로 검색할 수 있다.

검색	
설명	프로필이나 해시태그, 위치를 검색하는 기능을 가진 객체
필드	내가 검색한 목록 / 검색 목록 중 계정 / 해시 태그 / 위치
메소드	내 검색 목록 삭제 / 모든 검색 목록 보기 / 입력한 내용 검색 / 목록 표시

iv. 스토리 객체

게시물과 비슷한 특성을 가지고 있기 때문에 객체의 필드와 메소드에도 비슷한 부분이 있지만 확연히 단순했다. 스토리 특성상 게시 대상을 정할 수 있기 때문에 사생활 부분에서는 이점이 있었다. 게시물과 비슷하면서 다른 패러다임을 주고 있기 때문에 많은 사람들이 스토리를 이용한다.

스토리	
설명	해당 스토리의 게시자가 올린 스토리의 정보를 가지고 있는 객체
필드	게시자 / 게시 시간 / 태그된 사람 / 사진 / 내용 / 해시 태그 / 위치 / 스토리를 본 계정 / 스토리의 길이 / 게시 대상
메소드	다이렉트 메시지 보내기 / 스토리 신고 / 링크 복사 / 스토리 삭제 / 게시 대상 정하기

v. 다이렉트 메시지 객체

게시물, 스토리처럼 모든 사람을 대상으로 하는 것이 아니라 채팅방 구성원들만 공유하고 싶은 정보는 다이렉트 메시지를 이용한다. 다이렉트 메시지는 게시물, 스토리가 주 목적인 인스타그램의 특성상 메신저 기능이 주가 되는 다른 어플보다는 단순한 모습을 보였다.

다이렉트 메시지	
설명	인스타그램 이용자들끼리 메신저 기능을 구현한 객체
필드	채팅방 목록 / 채팅방에 있는 이용자 목록 / 채팅 목록 / 채팅한 시간 / 해당 채팅 읽음 / 읽지 않음 정보
메소드	메시지 전송 / 사진 전송 / 음성 전송 / 이모티콘 전송 / 음성 전화 / 화상 전화

4. Django Model 구현을 통한 객체 이해

객체를 이해하고자 하여 여러 검색을 해 보다 Django를 이용하여 블로그를 구현해보는 쉽고 편한 방법이 있다. Python 언어이지만 객체를 이해하기에는 좋은 방법인 것 같아서 블로그의 Model을 구현했다. 인스타그램의 모든 기능을 구현하지는 못했지만 객체 이해에는 도움이 되었다.

게시물 모델 구현

```
class Post(models.Model):
    author = models.ForeignKey(
        settings.AUTH_USER_MODEL, on_delete=models.CASCADE)
    title = models.CharField(max_length=200)
    content = models.TextField()
    image = ProcessedImageField(
        upload_to=post_image_path,
        processors=[ResizeToFill(600, 600)],
        format='JPEG',
        options={'quality': 90},
    )
    created_date = models.DateTimeField(
        default=timezone.now)
    published_date = models.DateTimeField(
        blank=True, null=True)
    like_user_set = models.ManyToManyField(
        settings.AUTH_USER_MODEL, blank=True, related_name='like_user_set',
through='Like')
```

```
    def publish(self):
        self.published_date = timezone.now()
        self.save()

    def __str__(self):
        return self.title

    def like_people(self):
        return self.like_user_set.all()

    @property
    def like_count(self):
        return self.like_user_set.count()

    def approved_comments(self):
        return self.comments.filter(approved_comment=True)
```

프로필 모델 구현

```
class Profile(models.Model):
    user = models.OneToOneField(
        settings.AUTH_USER_MODEL, on_delete=models.CASCADE)
    nickname = models.CharField(max_length=64)
    profile_photo = models.ImageField(blank=True)
    user_follow_set = models.ManyToManyField(
        settings.AUTH_USER_MODEL, blank=True, related_name='follow_user_set',
through='Follow')

    def follow_people(self):
        return self.user_follow_set.all()

    @property
    def follow_count(self):
        return self.user_follow_set.count()
```

인스타그램의 모델을 직접 볼 수 없어서 객체 파악에 답답함이 있었다. 부족한 기능이지만 Django를 통해서 비슷한 기능을 구현하고, 이를 변경하며 확인해보니 객체의 등록, 삭제, 변경 부분에서의 이해를 조금 더 쉽게 할 수 있었다.

5. 결론

처음으로 빠르게 성장하는 SNS 시장의 선두주자인 인스타그램 어플리케이션을 선정해서 기본 구조를 파악하고, 직접 쉬운 코드를 작성해보며 모델링 체험을 했다.

그 결과 객체를 파악하면서 사용자가 만족하는 어플리케이션을 설계하는 과정이 어렵고 고되다는 것을 알게 되었다. 하지만 완성도 높은 설계를 해 놓는다면 후에 사용자들의 피드백을 반영하기 쉽겠다는 생각을 하게 되었다.

여러 언어를 배우며 언어들의 이론을 배웠고, 코드를 작성해본 경험은 많았다. 하지만 이렇게 이미 선배 개발자들이 만들어 놓은 수준 높은 프로그램을 조금이나마 연구해보며 '설계'의 과정을 배운 것 같아 색다른 경험이었다. 이러한 과정을 학부 과정에서 여러 번 거치고 나면, 현업에 뛰어들었을 때 도움이 될 것이라고 생각한다.

Django 구현 코드 : https://github.com/LeeBumSeok/Practice-django_instagram