

기억을 기억하다

BackUP

Smile Developers aka SDS
이재빈 & 오민용



가족 앨범





An elderly woman with short brown hair is seated in a green wooden chair, looking down at a book she is holding. She is wearing a white patterned top with black and green geometric designs. The book's cover features a detailed illustration of yellow and orange flowers within an ornate, golden frame. The text on the book's spine is in Cyrillic script. The overall scene is softly lit, creating a calm and focused atmosphere.

하지만...

서비스 배경



대다수 SNS가 일회성 콘텐츠



단순한 사진·동영상 공유 위주의 서비스

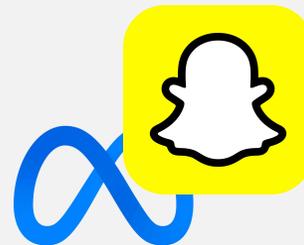


SNS, 모든 연령대가 쉽게 접근가능하기 어려움

서비스 배경



대다수 SNS는 일회성 콘텐츠



서비스 배경



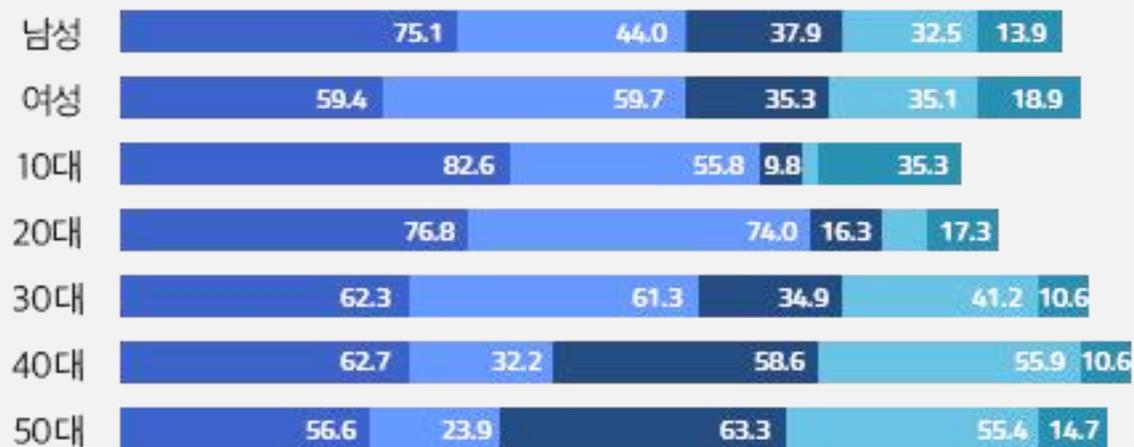
단순한 사진·동영상 공유 위주 서비스



SNS, 모든 연령대가 쉽게 접근가능하기 어려움

성·연령별 SNS 이용 서비스

(단위 : %, 중복 선택)



■ 페이스북 ■ 인스타그램 ■ 밴드 ■ 카카오토리 ■ 트위터

[자료 제공 = 나스미디어]

서비스배경



SNS, 모든 연령대가 쉽게 접근가능하기 어려움

성·연령별 SNS 이용 서비스

(단위 : %, 중복 선택)



10 ~ 30대 - 페이스북, 인스타그램

40 ~ 50대 - 밴드, 카카오토리

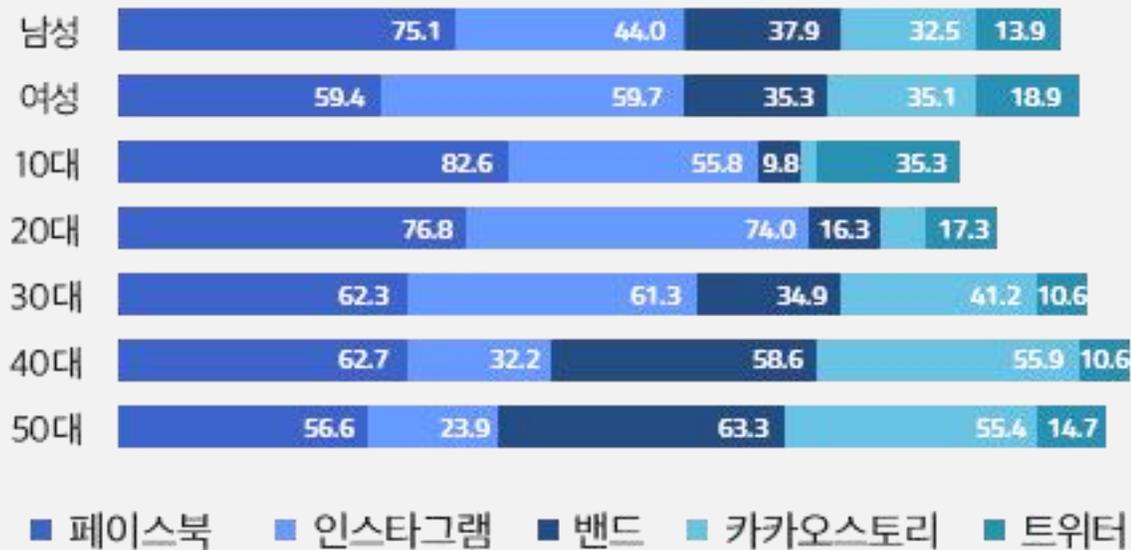
서비스배경



SNS, 모든 연령대가 쉽게 접근가능하기 어려움

성·연령별 SNS 이용 서비스

(단위 : %, 중복 선택)



10 ~ 30대 - 페이스북, 인스타그램

40 ~ 50대 - 밴드, 카카오토리



니즈가 다름

일회성 콘텐츠에 익숙

+

디지털 세대차이



가족 & 집단의 이벤트 보관X

기존 서비스의 문제

	공유	앨범	SNS	검색	재접근성
 Google Photos	✓	✗	✗	✓	△
 네이버 밴드	✓	△	✓	△	△
 삼성 갤러리	△	✓	✗	✗	△
 iCloud	✓	✗	✗	✓	△

새로운 앨범 플랫폼

주요기능



랜덤 알림으로 가끔씩 옛 앨범 돌아보기



타임 캡슐로 더 재미있는 사진 공유



썸네일과 카테고리로 깔끔한 앨범 관리



최소한의 기능으로 쉬운 사용

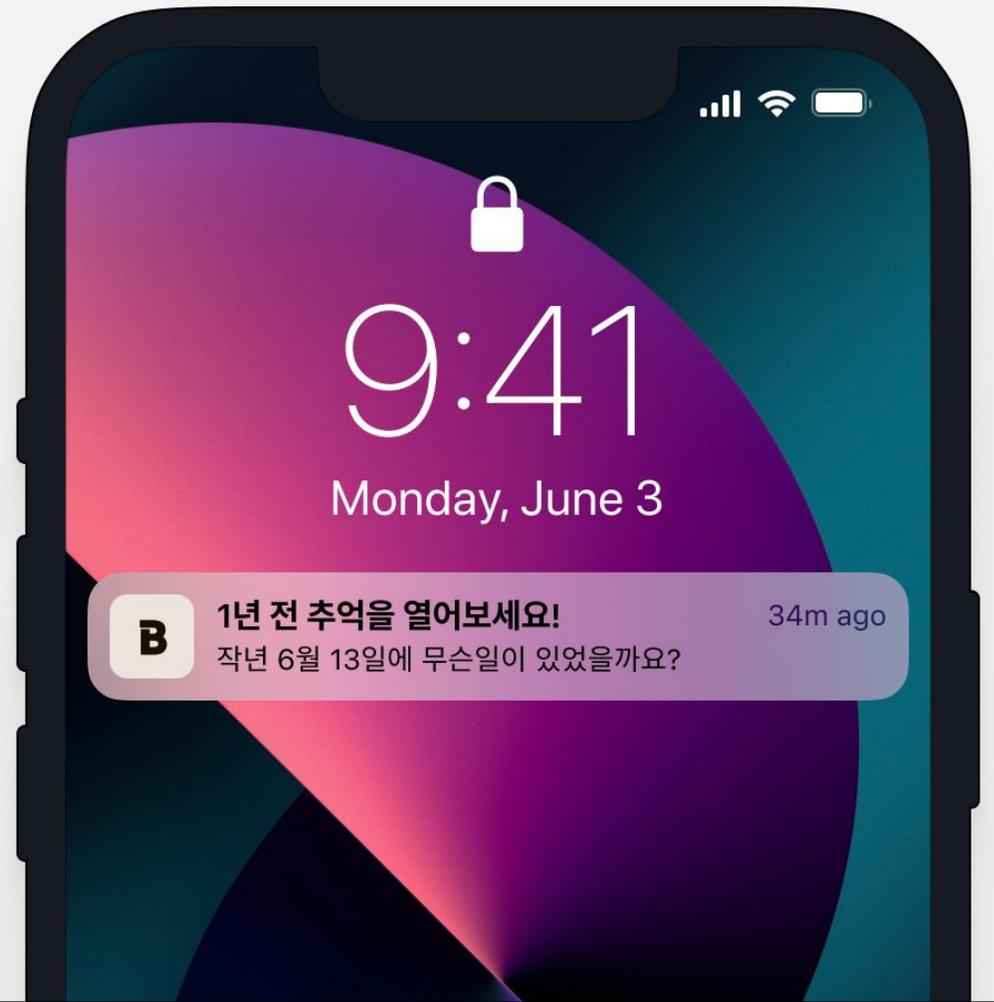


추억 되돌아보기

속제같이 하던 업로드는 그만,

가끔씩 **지난 추억**을 보는 재미가 있어야죠 :)

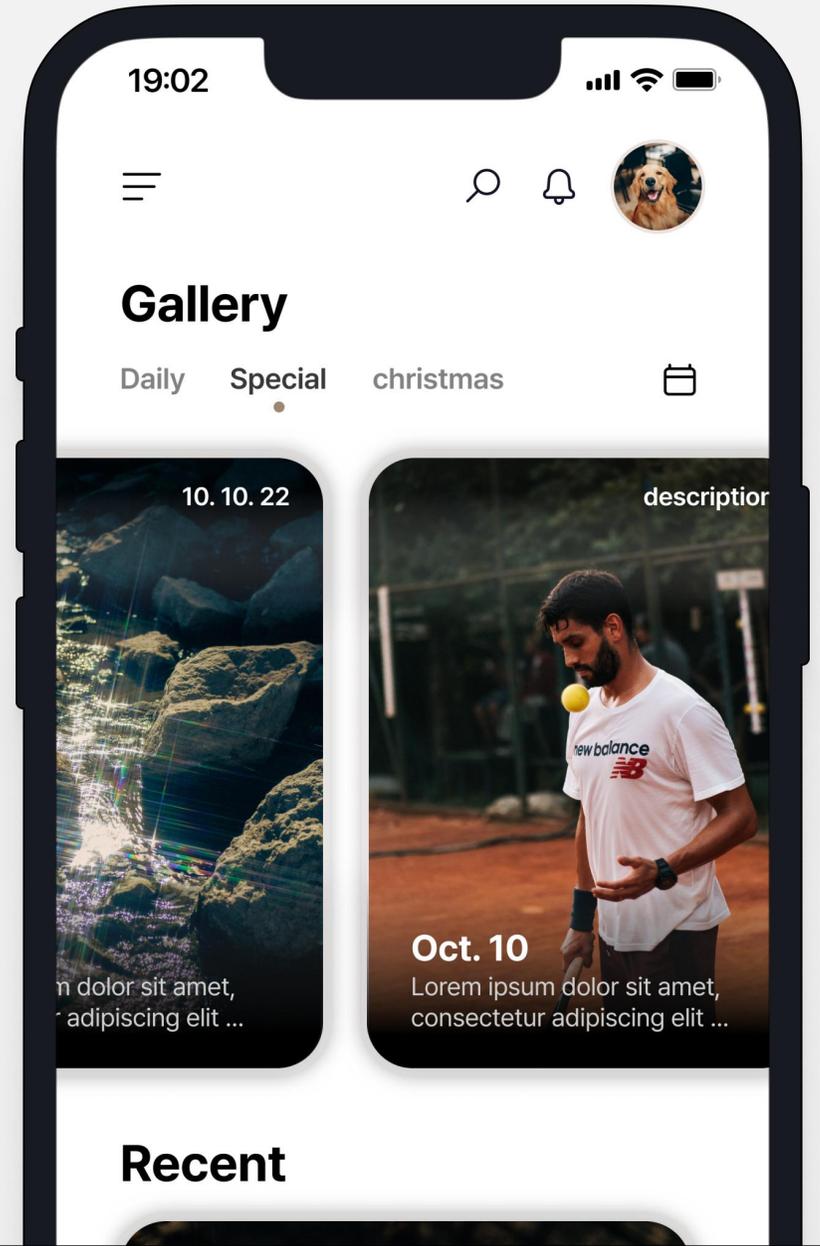
짧게는 한달, 길게는 1~3년간격으로 랜덤 앨범을 푸시알림으로 추천해줍니다.



많아질수록 찾기 어려워지는 앨범들,

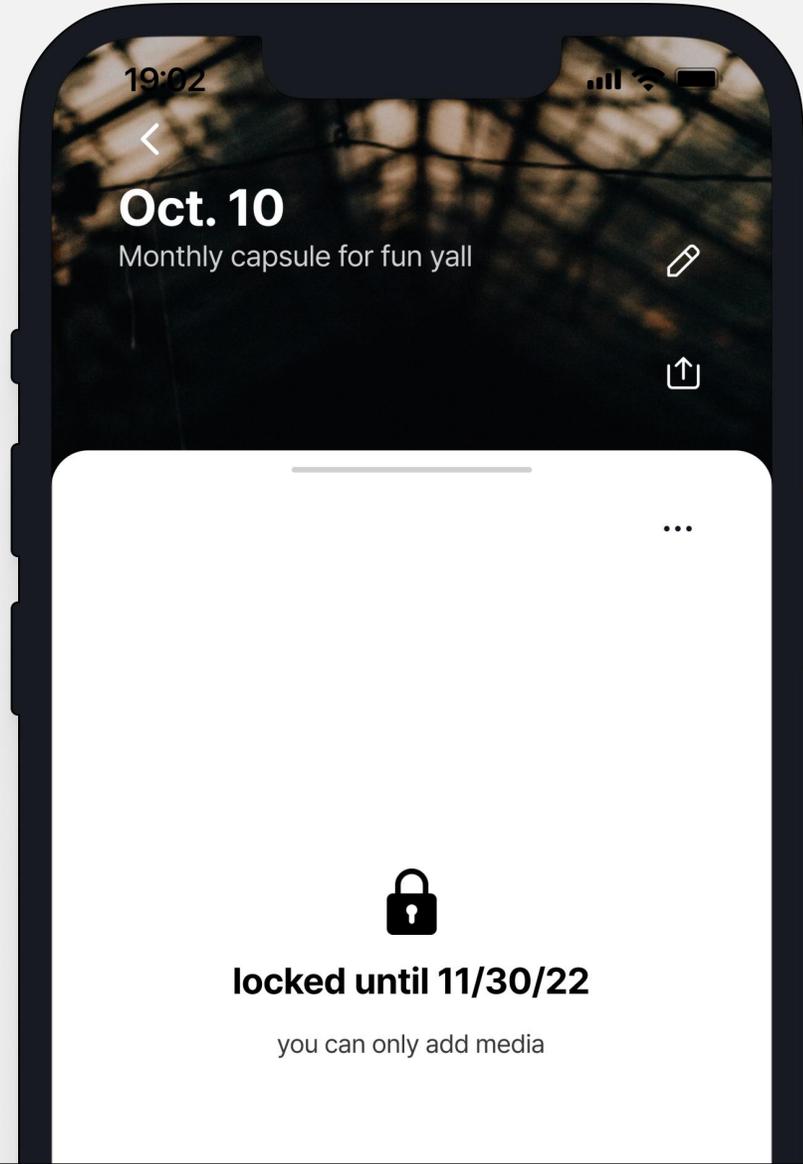
썸네일과 카테고리로 깔끔하게 정리해봐요

제목뿐 아니라 앨범 설명, 이벤트 날짜, 기여자(미디어 업로드 유저), 썸네일을 추가할 수 있습니다.





타임캡슐



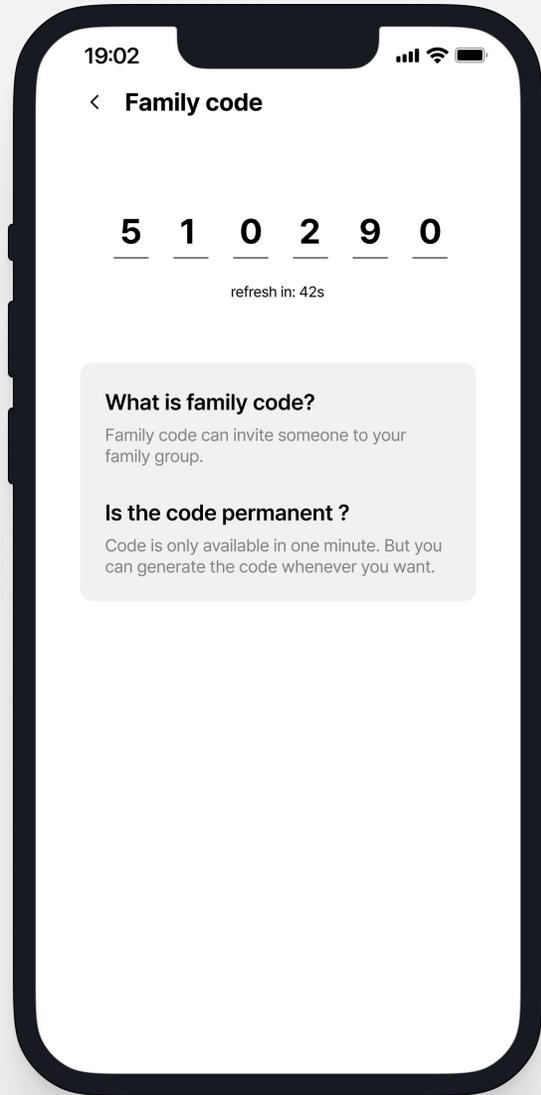
언제까지 단순한 업로드만 할건가요?

타임캡슐로 더 재미있게 미디어를 공유해봐요!

타임캡슐은 각자 사진을 업로드 하고 특정 시간부터 열람할 수 있는 특수한 형태의 앨범입니다.



간단한 UI-1



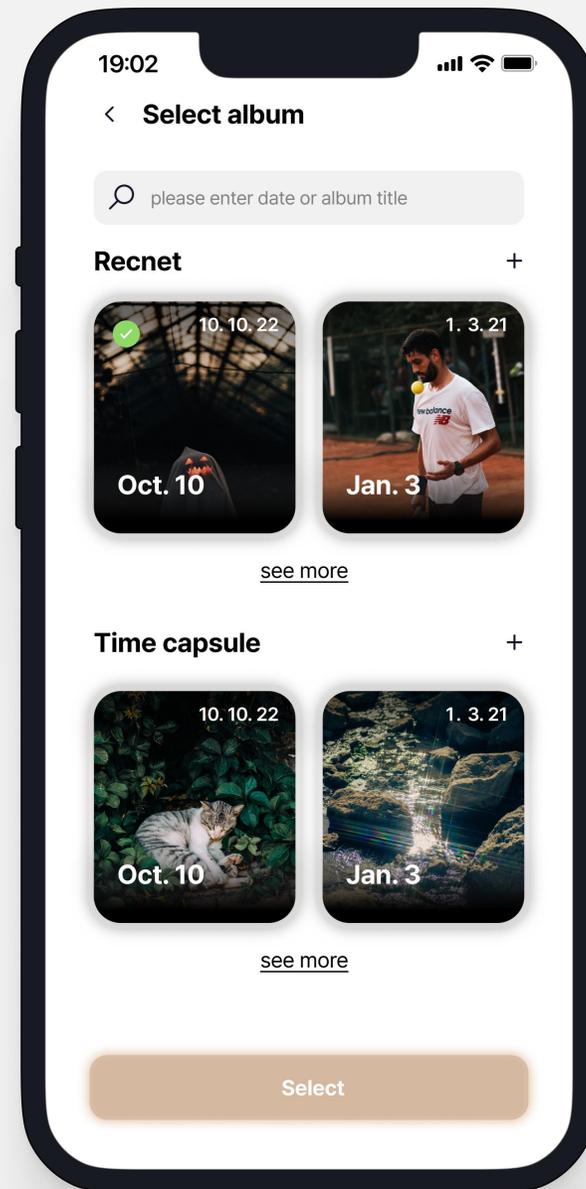
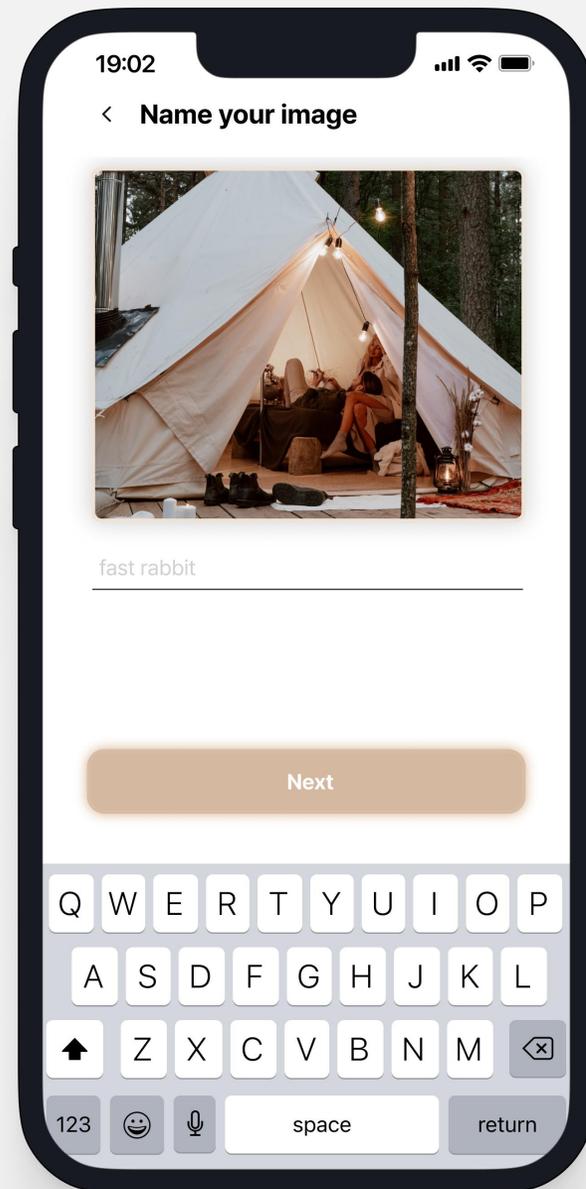
패밀리 코드 입력만으로 초대 가능

이 코드는 온보딩에서 사용할 수 있습니다.
패밀리 코드는 1분 동안만 유효하며, 영구적인
발급이 아닙니다.



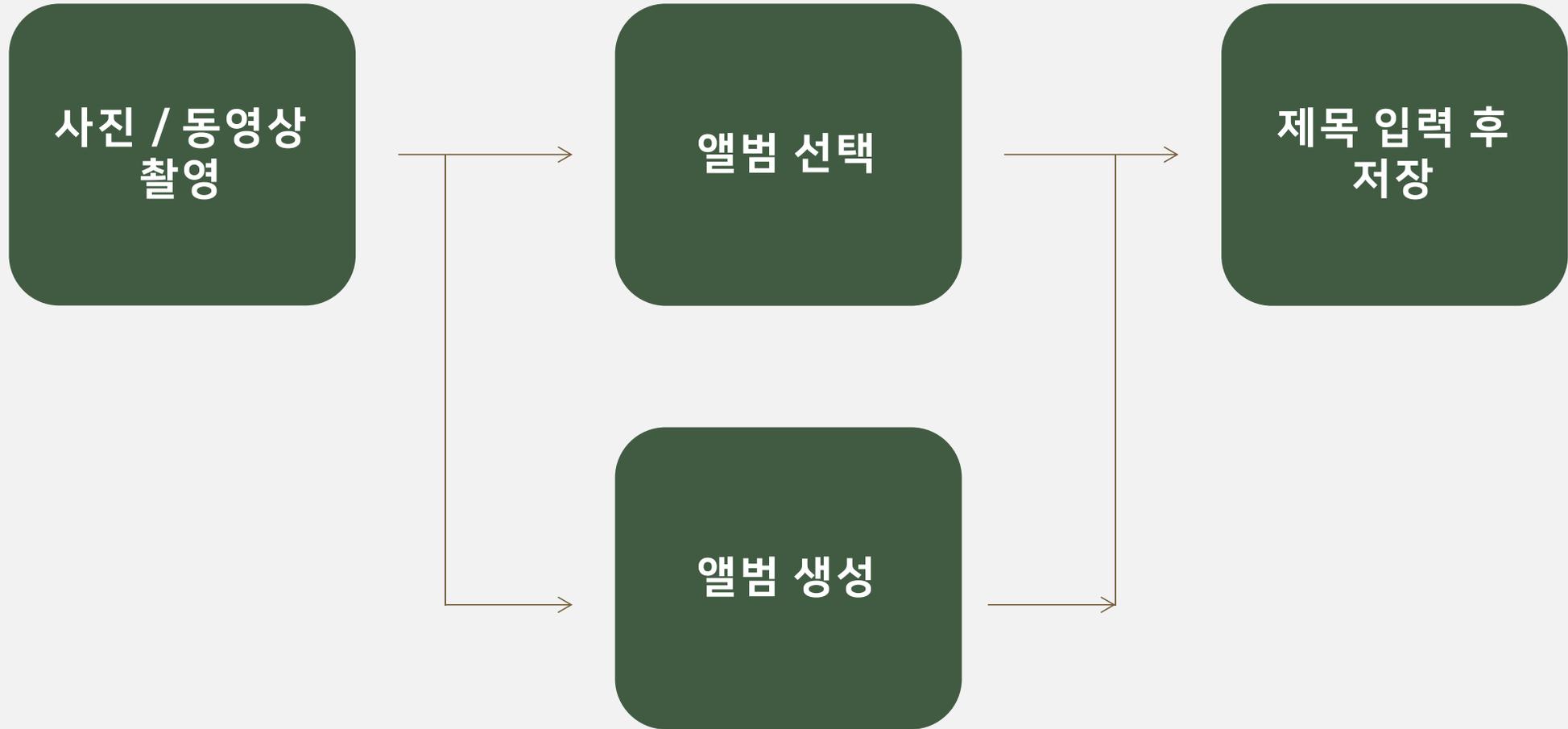
간단한 UI-2

업로드 방식 간소화



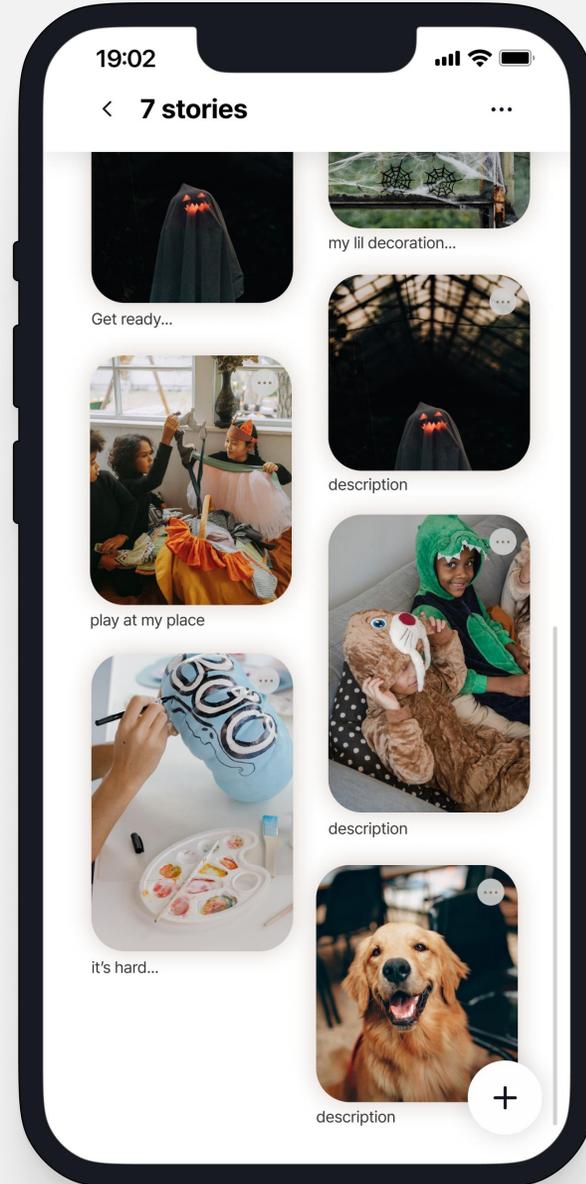
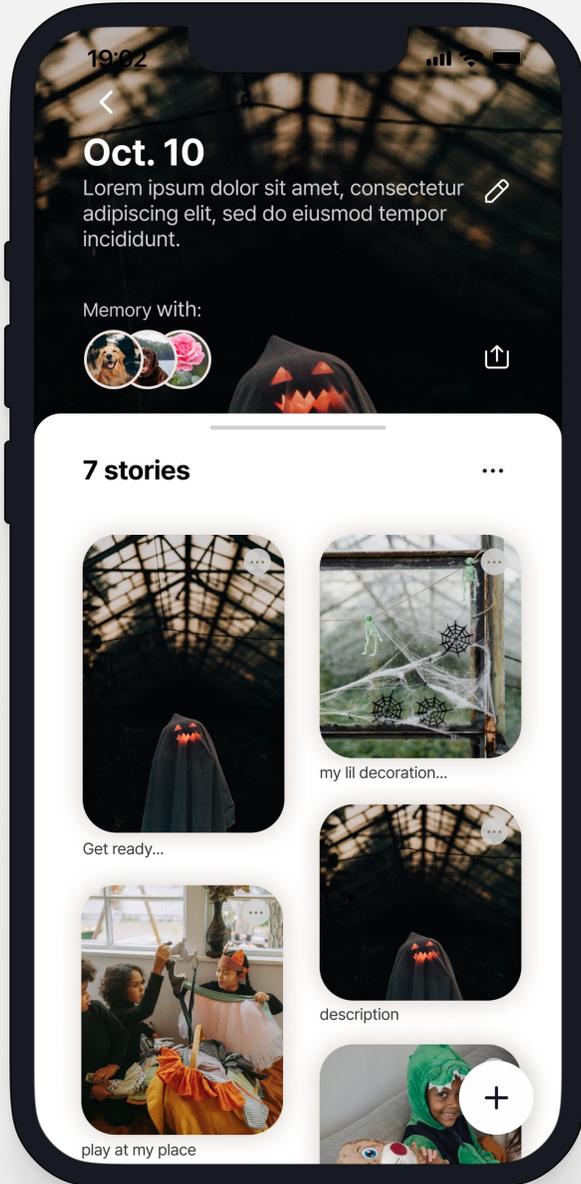


간단한 UI- 업로드 간소화





간단한 UI-2

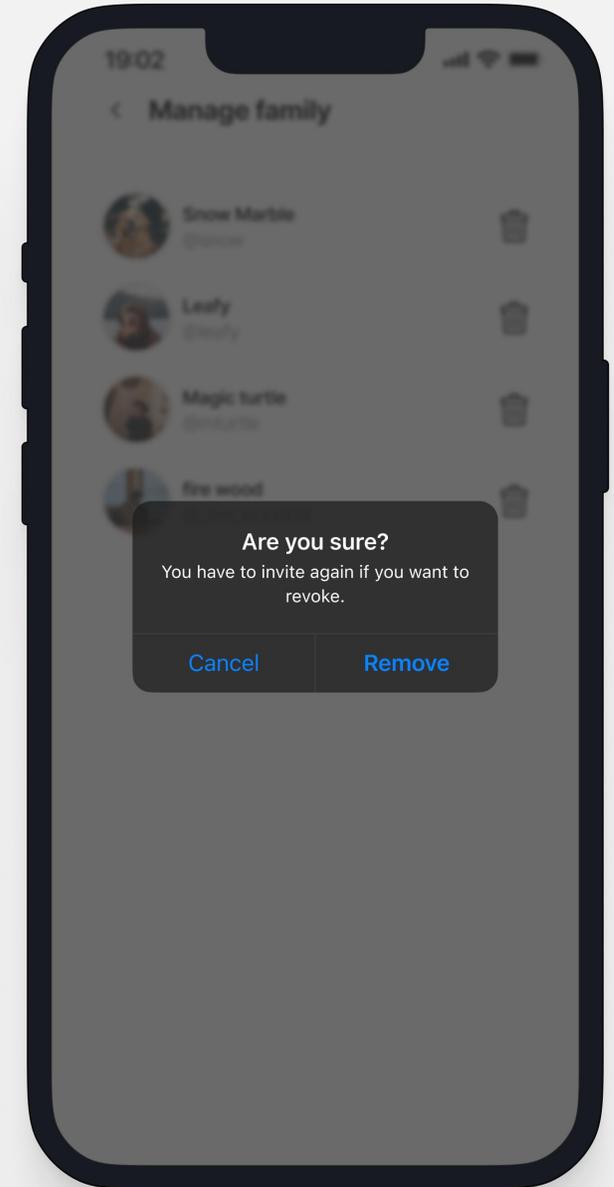
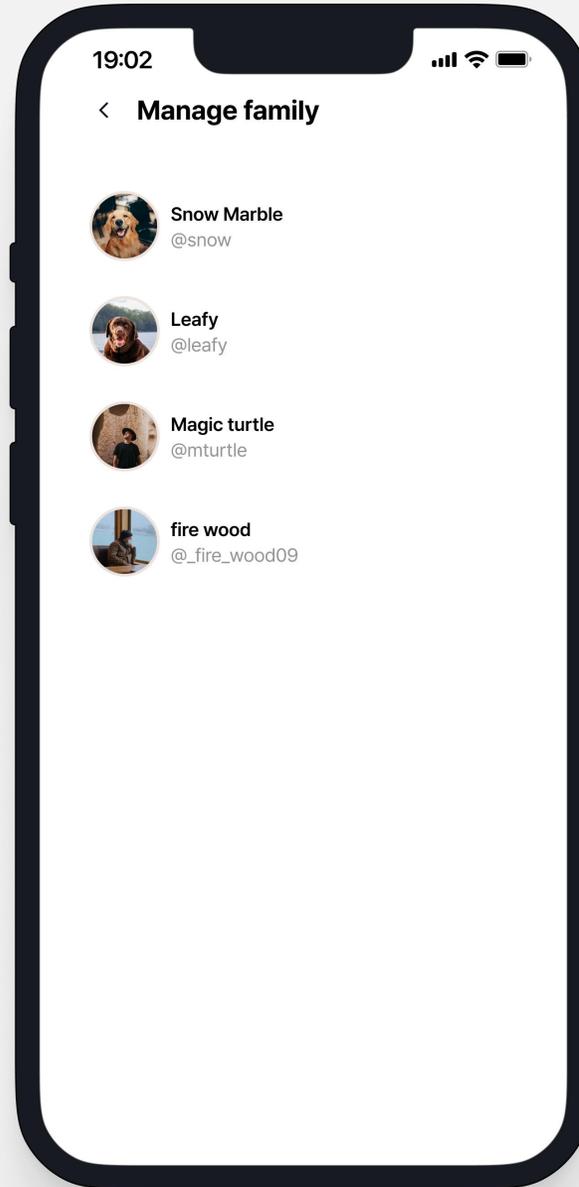


앨범 디테일과 직관적인 UI



간단한 UI-2

멤버 관리



서비스 개발의 여정

개발과정

아이템선정 [~10/30]

- 아이템 회의 및 기획서 작성 중심 회의
- 시장 분석, MVP 선정

Notion

기획

v1

가족 앨범의 접근성이 떨어졌다는 것이 아이디어의 시작이다. 현재 시중의 서비스는 사진이나 영상 같은 미디어를 공유하는 것에 그치기 때문에 각각의 이벤트를 따로 관리하거나 찾아보기 쉽지 않다. 기존의 물리적 앨범의 불편한 점을 크게 해소할 수 있는 디지털 앨범 관리를 기반으로 접근성과 사용 편의성을 높여주는 서비스를 기획한다.

제안 배경

현재 사용되고있는 대다수의 SNS에서 사진 공유 아이디어가 시작되었다. 자신의 최근 일을 공유하는 이벤트에서도 이러한 일회성이 작용한다. 현재 SNS 밴드는 미디어를 공유하는 폐쇄형 SNS가 많아 접근성이 떨어지는 점이 있어 일회적 소비의 문제점으로 인식되고 있다. 또한, SNS 서비스로 인한 다소 복잡한 UI와 MZ세대와 밀접한 관련이 있는 SNS 서비스로 인해 대 이상이 11%이다.

현재 서비스의 문제점

현재 널리 쓰이는 그룹 내 미디어 공유 앱

- Google Photos
- Naver Band
- Samsung Gallery(Shared album)
- Dropbox
- i-Cloud

Google Photos

장점

- 앨범 생성 및 공유 가능
- 구급의 이미지 검색 기능

Figma

PAAS-TA - 백업(Back-Up)

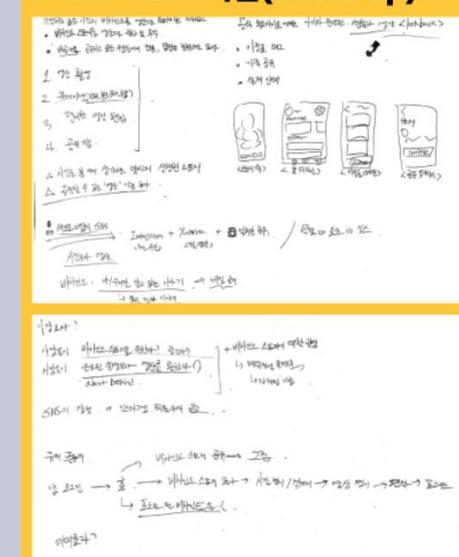
Community 기능 느낌으로 팔로우 없는 SNS 개발 => 밴드

사진마다 1분 길이의 영상 몇개인데

그룹화, 사진이 폴더가 되는거야 사진 안에 영상이 있을수도 있고 공유한 사람들끼리 또 수정이 가능해

디자인도 책 넘기는 것처럼

Community 기능 느낌으로 팔로우 없는 SNS 개발 => 밴드



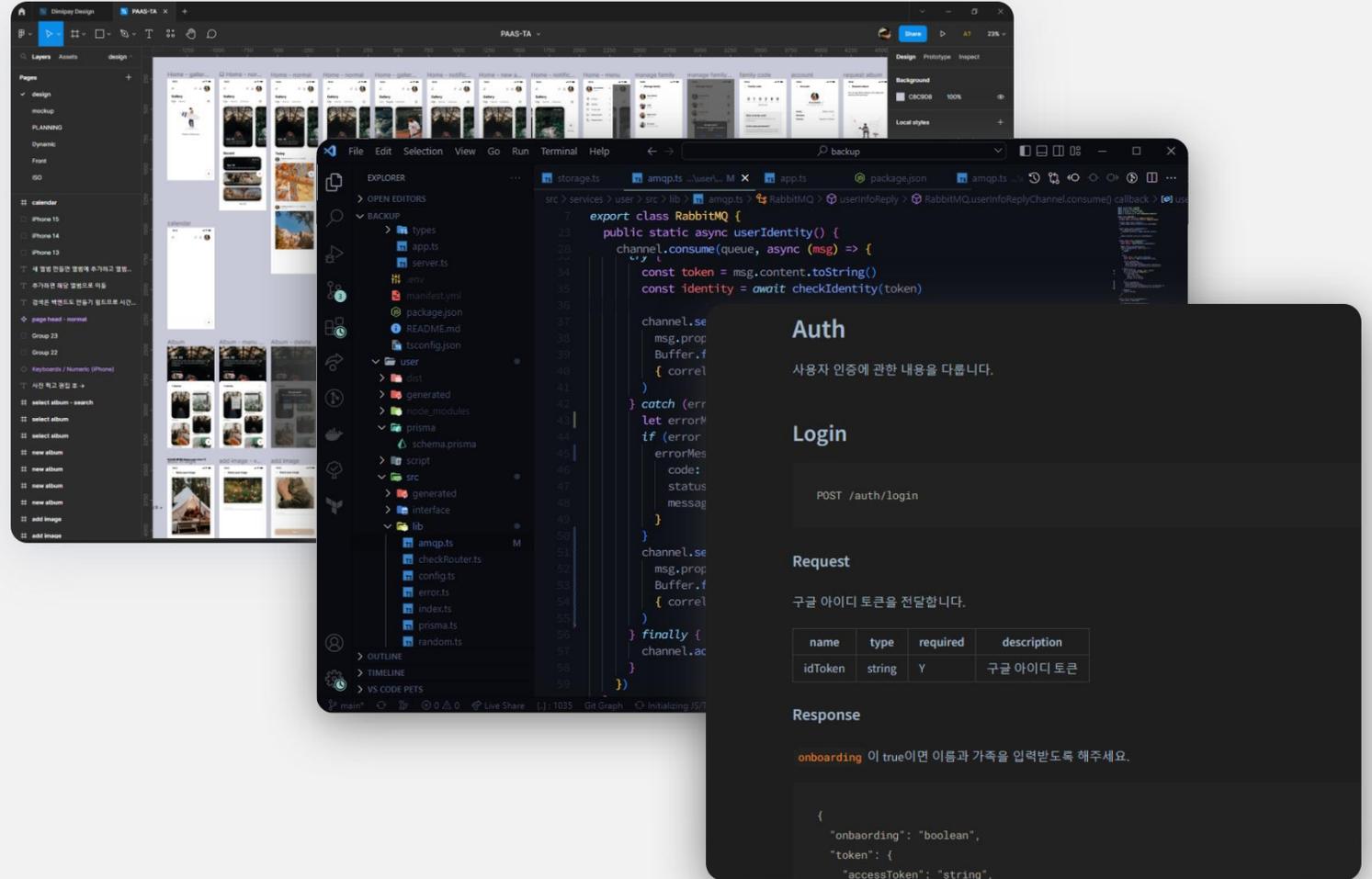
- 앱에 들어가서 카메라로 사진을 찍거나 동영상 찍어오. a. 위의 오버레이로 메뉴 이동
- 동영상일 경우 간단한 편집을 진행합니다.
- 모든 과정이 끝난 후 글을 적거나 앨범을 지정해 줘요.
- 공유할 그룹을 선택해요.
- 마지막으로 SNS에 올릴지 선택해요. a. SNS에 올리는 것은 앨범 하나
-

앱 시작 → 카메라 오픈 → 동영상 및 사진 찍기 → 편집 및 업로드

개발과정

아이템구체화및개발 [~11/12]

- 앱 디자인과 아키텍처 설계 시작
- 앱, 백엔드 동시 개발
- 두 차례의 긴급 회의
- API문서 작성 및 API 연동
- 상세 기능 구현



개발과정

w/이승윤M

멘토링 [10/25, 11/15]

- 파스타 환경 개발 교육
- 앱 방향성과 아이디어 검토
- 기획 및 아이디어 피드백
- 발표 구성 멘토링

PAAS-TA 멘토링 - 2

공유 앨범 서비스인데 백업인데 조금 헷갈렸다... 저장소 느낌이 너무 강하다. 고전적으로 가보는게 어떨까...?

1. 저희 서비스 아이디어

- 제안 배경이 좀 있고, ex) 수치상으로 있는지.
- 추구하는 방향
- 다른 서비스와
- 클라우드 저장
- 방향성을 재미 느낌이 없다 /
- 기능들이 공유
- 사회적 책임? 풀어나가야 할

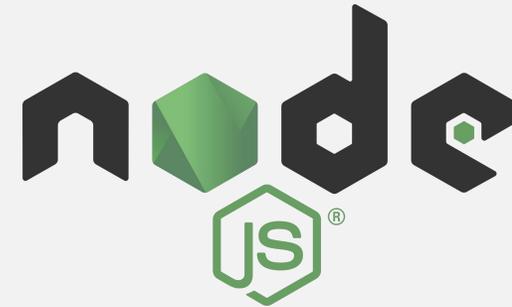
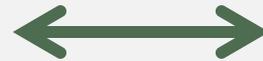
2. 서비스 개발 배경에
이나 불편했던 경험 등

- 찾는게 어려웠
- 레시피 서비스
- 인공지능이 찾
- 태그 기능이라

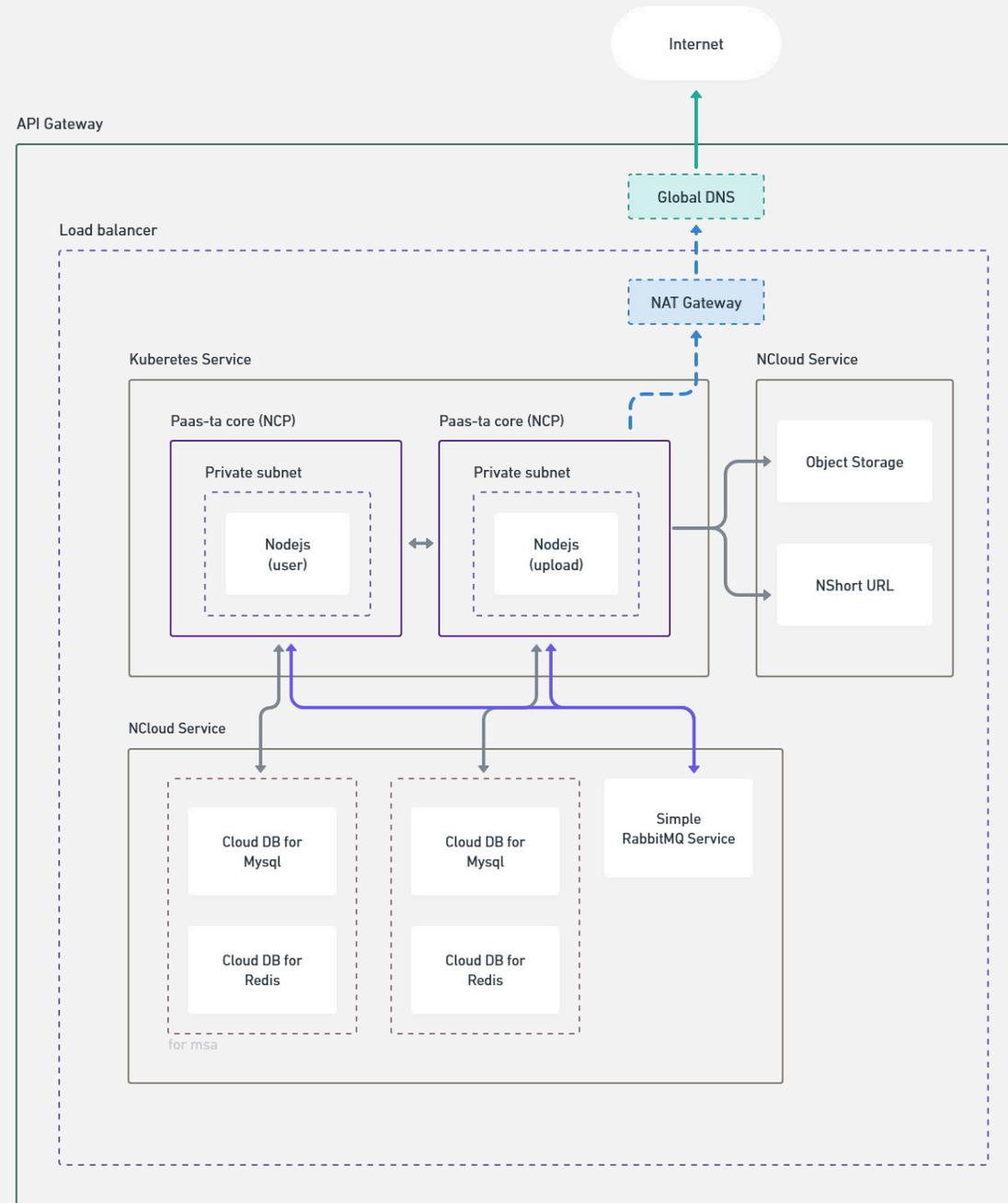
서비스 구조 (전체)



Flutter

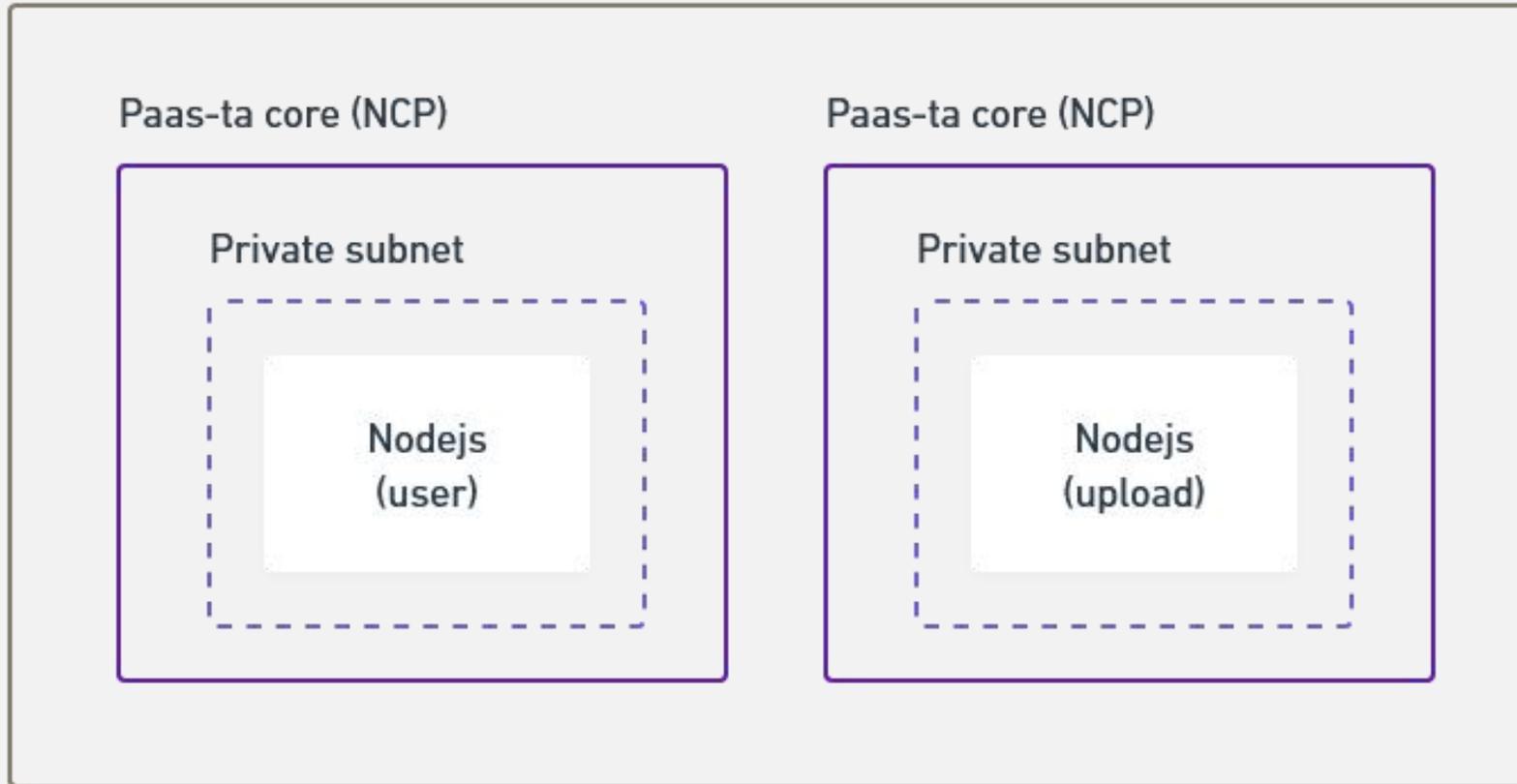


서비스 구조 (백엔드)



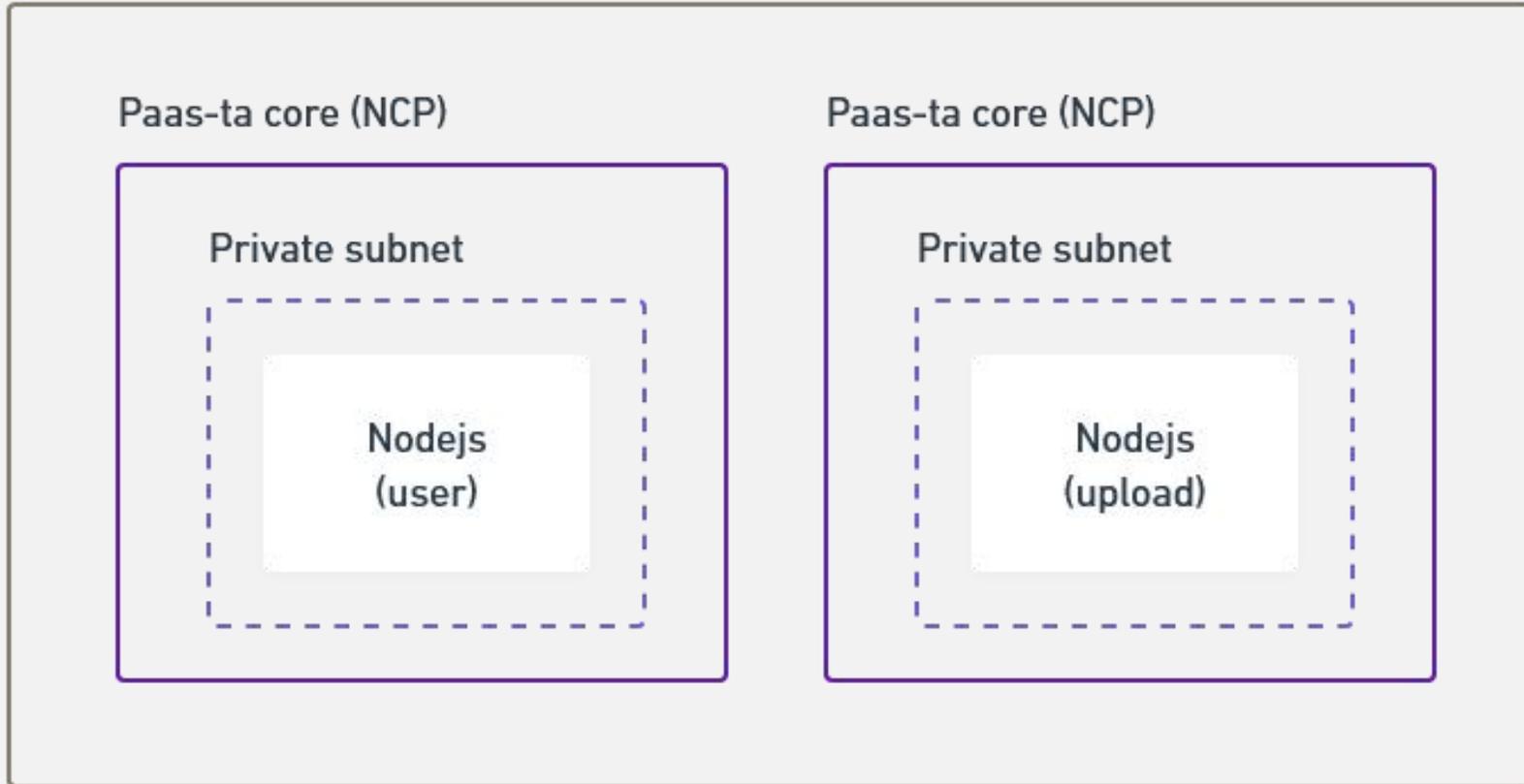
서비스 구조 (백엔드)

Kuberetes Service



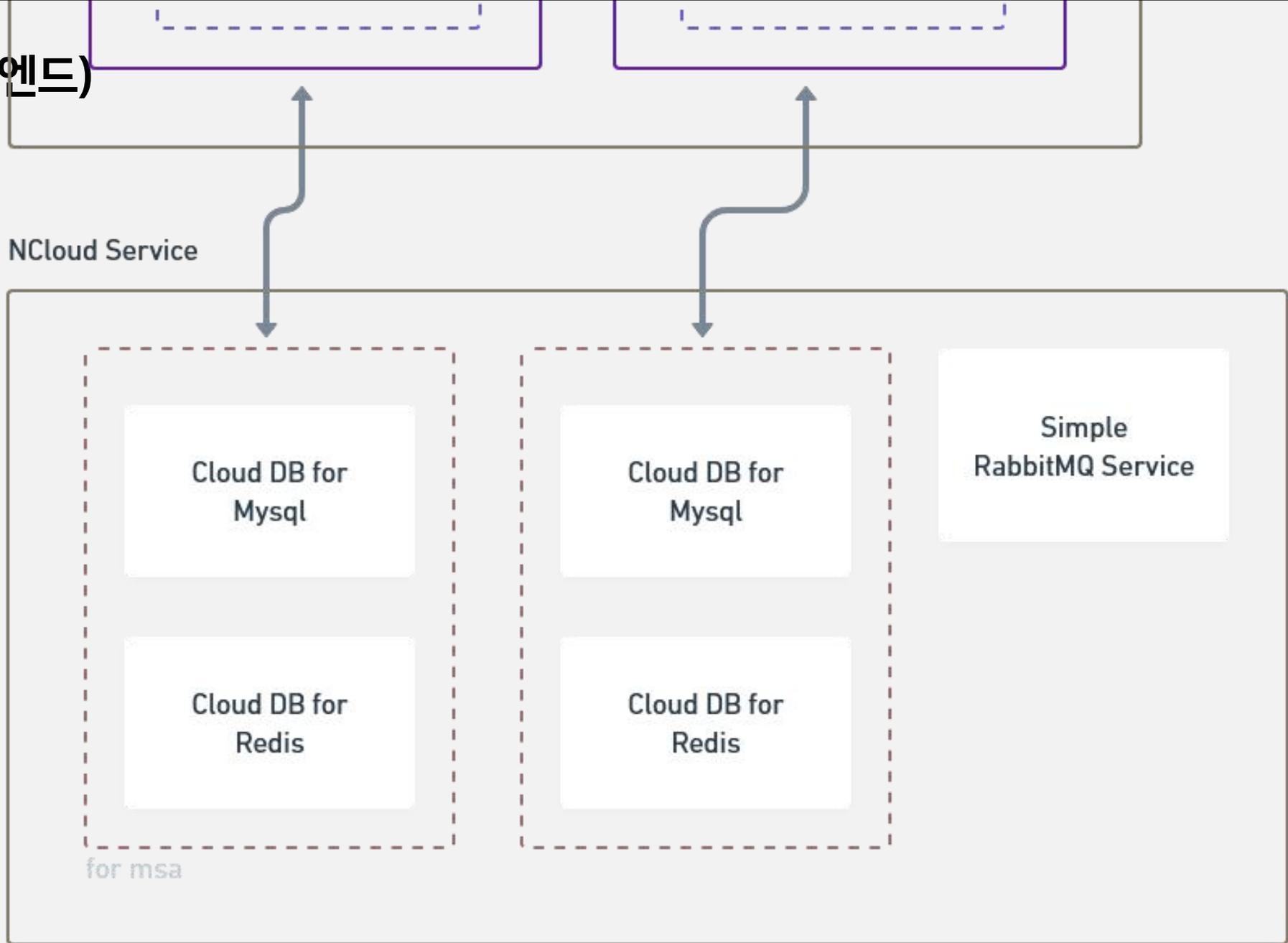
서비스 구조 (백엔드)

Kuberetes Service



```
src\services
├── upload
│   ├── dist
│   ├── generated
│   ├── node_modules
│   ├── prisma
│   ├── script
│   └── src
│       ├── .env
│       ├── manifest.yml
│       ├── package.json
│       ├── README.md
│       └── tsconfig.json
├── user
│   ├── dist
│   ├── generated
│   ├── node_modules
│   ├── prisma
│   ├── script
│   └── src
│       ├── generated
│       ├── interface
│       ├── lib
│       ├── middlewares
│       ├── routers
│       ├── types
│       ├── app.ts
│       └── server.ts
│       ├── .env
│       ├── manifest.yml
│       ├── package.json
│       ├── README.md
│       └── tsconfig.json
```

서비스 구조 (백엔드)



NCloud Service

Cloud DB for
Mysql

Cloud DB for
Redis

for msa

Cloud DB for
Mysql

Cloud DB for
Redis

Simple
RabbitMQ Service

서비스 구조 (백엔드)



Prisma

type -safety & auto-completion

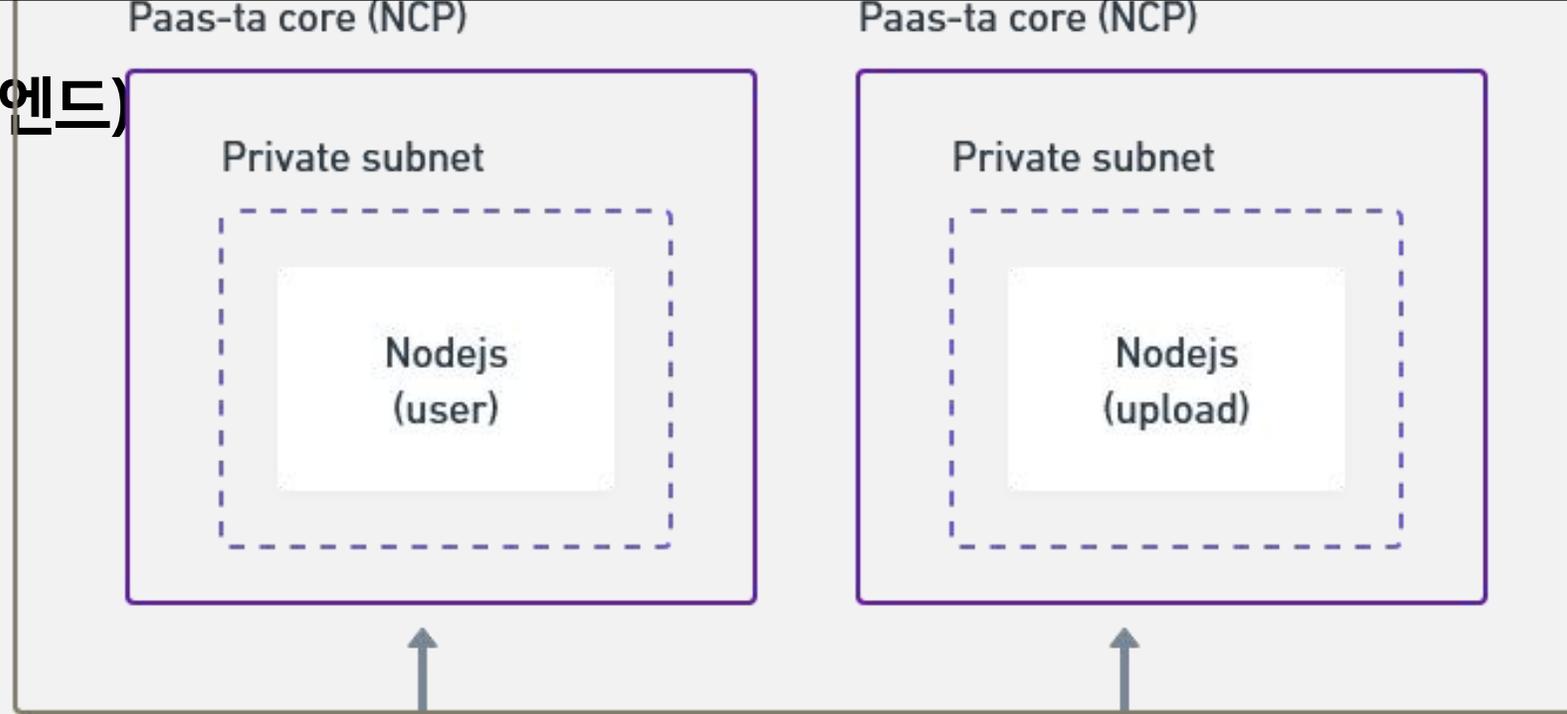
```
model Category {
  id      Int    @id @default(autoincrement())
  name    String
  familyId Int

  createdAt DateTime @default(now())
  updatedAt DateTime @default(now()) @updatedAt

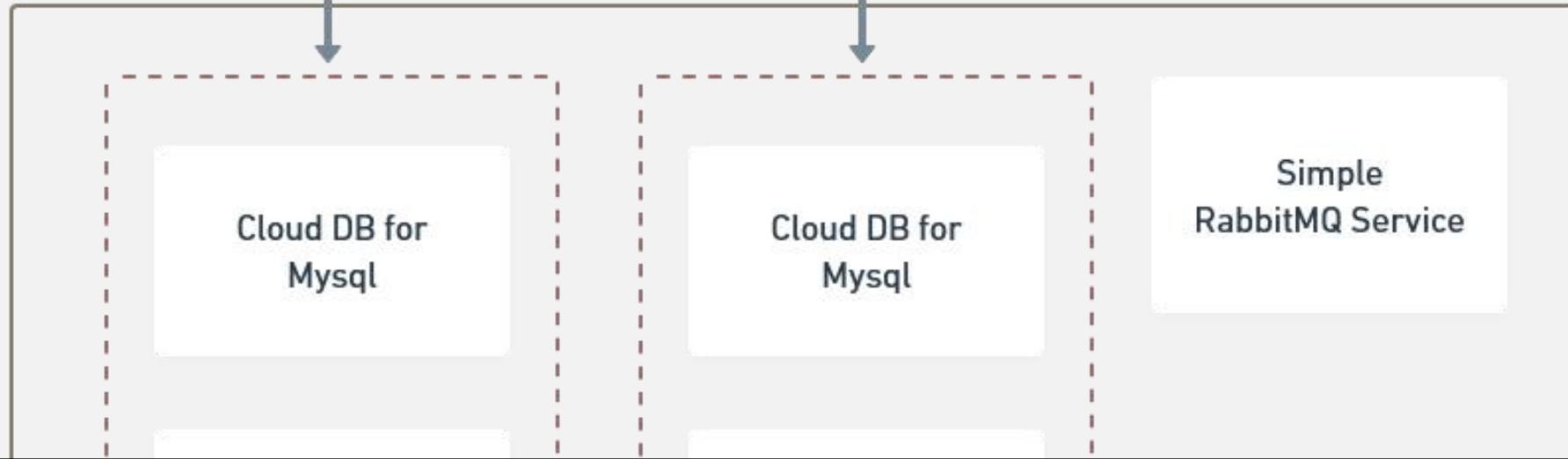
  albums Album[]
}

model Story {
  id          Int    @id @default(autoincrement())
  userId      Int
  description String?
  image       String
  createdAt   DateTime @default(now())
  updatedAt   DateTime @default(now()) @updatedAt
  lastViewed  DateTime @default(now())
  familyId    Int
  AlbumId     Int
  Album       Album  @relation(fields: [AlbumId], references: [id], onDelete: Cascade)
}
```

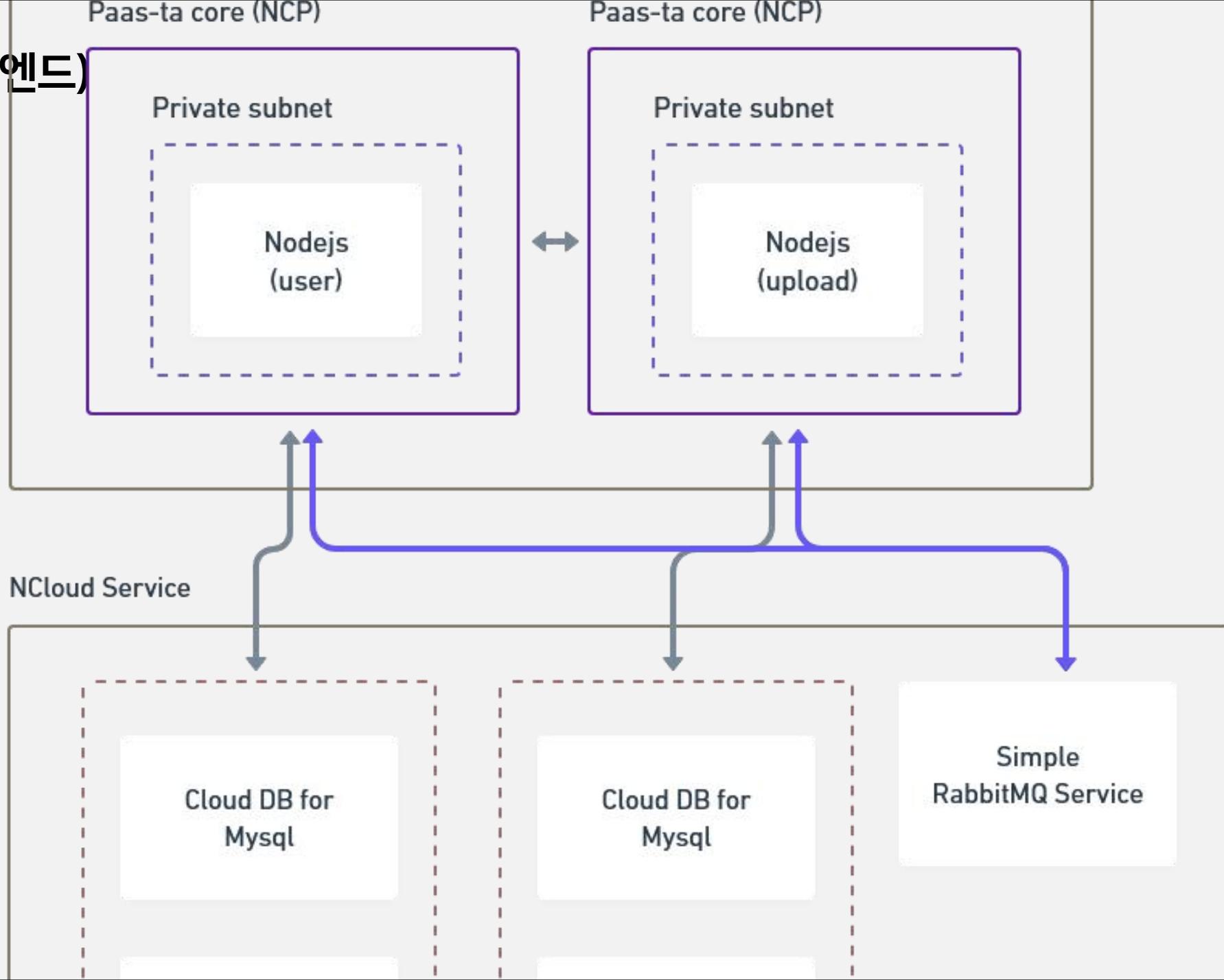
서비스 구조 (백엔드)



NCloud Service



서비스 구조 (백엔드)



Paas-ta core (NCP)

서비스 구조 (백엔드)

Private subnet

Nodejs
(upload)

Cloud DB for
Mysql

Simple
RabbitMQ Service

```
export const userInfo = async (userId: number) => {
  const queue = "user-info"
  const replyQueue = "user-info-reply"

  if (!userInfoChannel) {
    userInfoChannel = await RabbitMQ.createChannel()
    await userInfoChannel.assertQueue(queue)
    await userInfoChannel.assertQueue(replyQueue)
    await userInfoChannel.consume(replyQueue, (msg) => {
      if (!msg) {
        return
      }

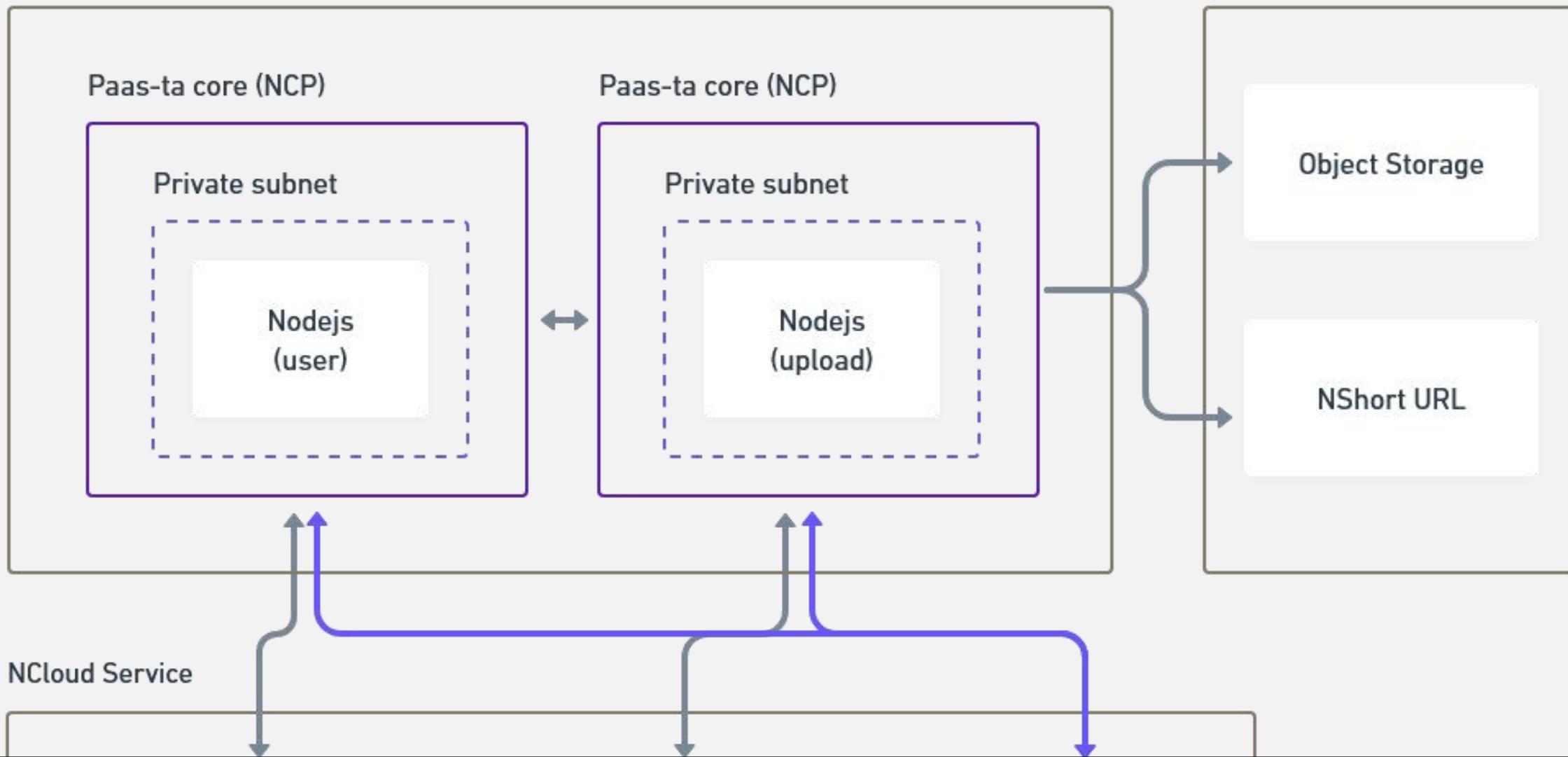
      const info: User = JSON.parse(msg.content.toString())
      const handler = userInfoHandler.get(msg.properties.correlationId)
      handler && handler(info)
      userInfoChannel.ack(msg)
    })
  }

  const correlationId = crypto.randomBytes(16).toString("hex")
  const promise = new Promise<User>((resolve) => {
    userInfoHandler.set(correlationId, resolve)
  })
  userInfoChannel.sendToQueue(queue, Buffer.from(userId.toString()), {
    correlationId,
    replyTo: replyQueue,
  })
  return promise
}
```

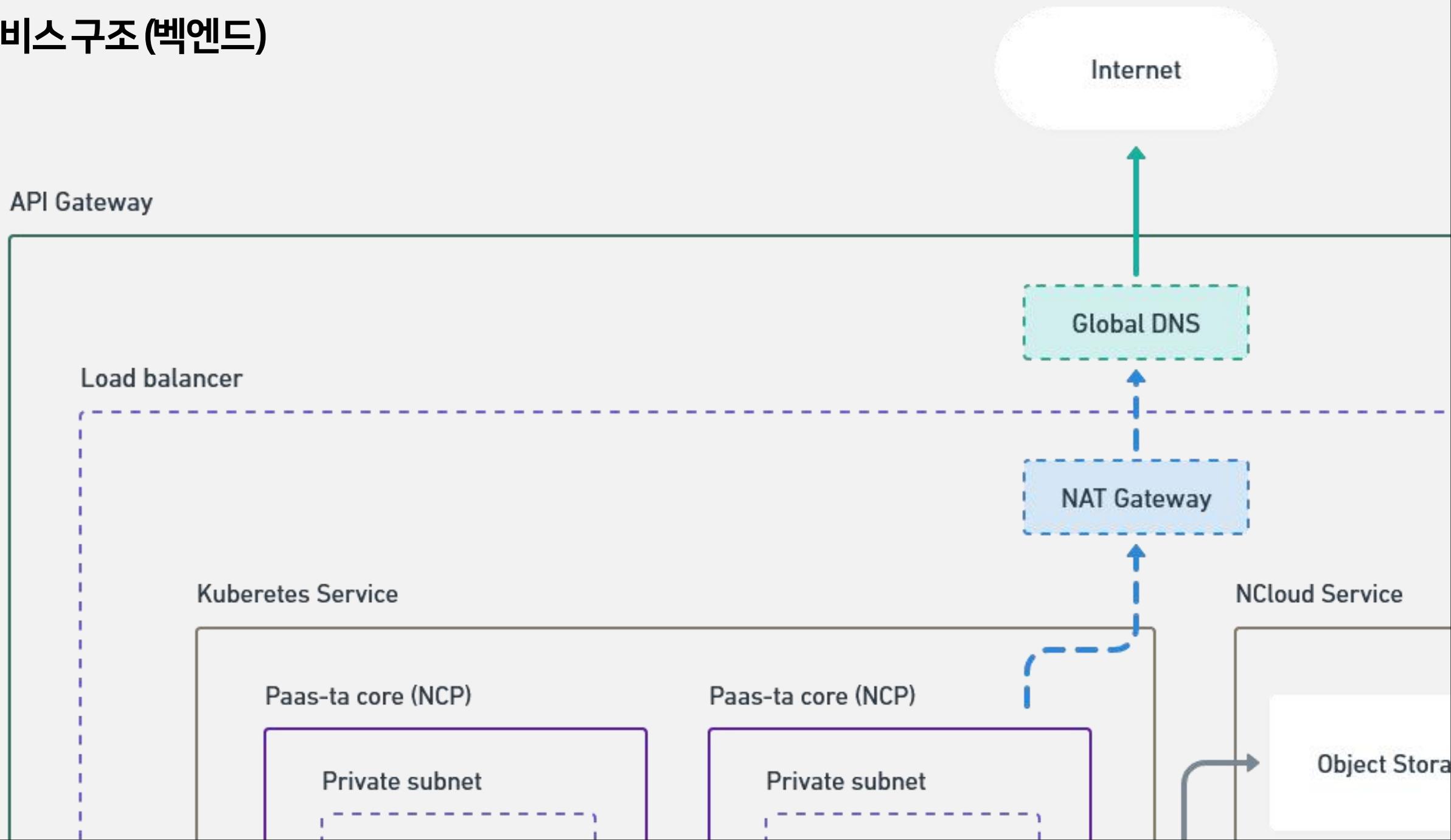
서비스 구조 (백엔드)

Kuberetes Service

NCloud Service



서비스 구조 (백엔드)



서비스 구조 (백엔드)

API Gateway

Load balancer

Kuberetes Service

Paas-ta core (NCF

Private subnet

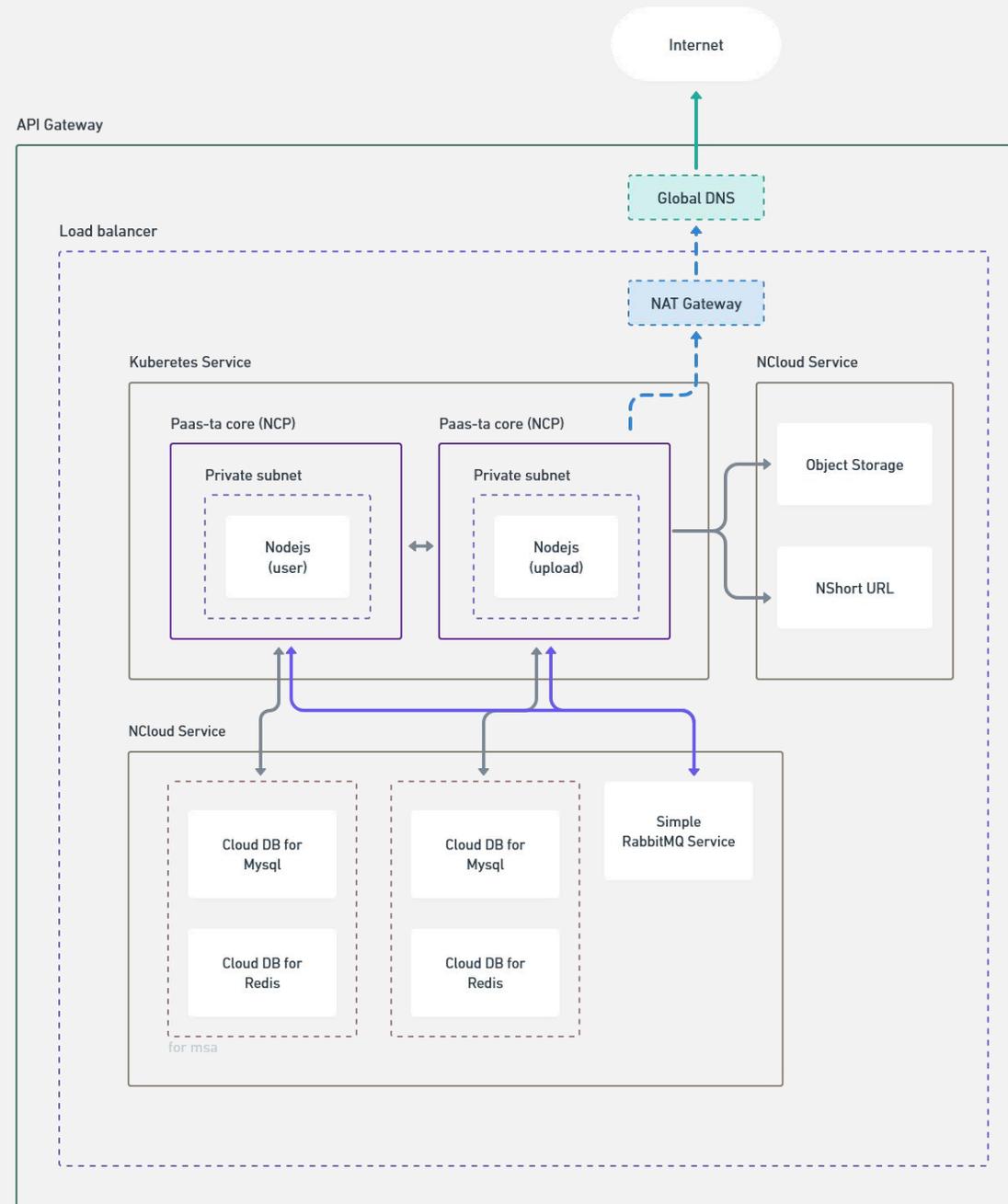
The screenshot shows a console interface for an API Gateway. At the top, it displays 'sample (9ssjw83u9y)' and navigation tabs: Resources, Stages, Gateway Responses, Models, and Overview. Below these are buttons for '리소스 생성', '리소스 보기', '리소스 삭제', and 'API 가져오기'. The main content area is divided into two panels. The left panel shows a tree view of resources under the root path '/', with the '/album' resource selected. The right panel shows the configuration for the selected '/album (701f4c5xd3)' resource. It includes a '메서드 생성' dropdown menu and two tables for GET and OPTIONS methods. Each table lists properties like '엔드포인트', 'API Key 필요', '인증', and '유효성검사'.

GET	보기	삭제	OPTIONS	보기	삭제
엔드포인트	GET /album		엔드포인트	OPTIONS /album	
API Key 필요	아니오		API Key 필요	아니오	
인증	NONE		인증	NONE	
유효성검사	NONE		유효성검사	NONE	

ice

ct Storage

서비스 구조 (백엔드)



서비스 구조 (백엔드)

CLOUD FOUNDRY

서비스 구조 (백엔드)

```
---
applications:
  - name: upload
    path: ./dist
    buildpacks:
      - paketo-buildpacks/nodejs
    command: npm run db:push && node -r tsconfig-paths/register server.js
    env:
      TS_NODE_BASEURL: ./
      NODE_PATH: ./
      JWT_SECRET: bc3cfdd021217a2d3ee62bcd7739506b
      UPLOAD_DATABASE_URL: mysql://***:***.gov-ntruss.com:3306/backup
      RABBITMQ_URI: amqp://***:***.vrmq-gov.naverncp.com:5672
      UPLOAD_REDIS_URI: redis://***:3***.gov-ntruss.com:6379
      NCP_ACCESS_KEY: ****
      NCP_SECRET_KEY: ****
      PRISMA_OUTPUT: ../generated/client
```

서비스 시연

기대 효과

일회성이 아닌 다회성



일회적 소모가 아닌 서로의 경험을 기억하는 서비스 제공

접근성 확대



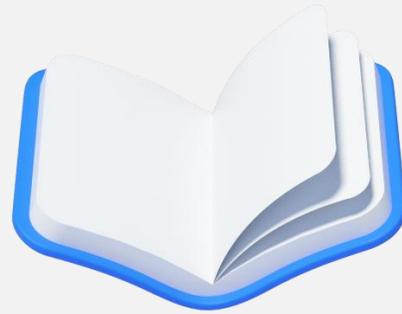
모두가 쓰기에 적합한 유니버설 디자인 적용

경험 및 기억 공유



사용자의 진실된 경험을 기억하며 진실된 가치 공유

물리적 앨범 한계 극복



다음 여정

다음 여정



개발보수및인정화

- 불안정한코드및구조리팩토링
- 기존기능보수
- 디자인및사용플로우개선
- 배포환경개선



추가기능개발과배포

- 추가기능 구현
- 추가기능 디자인추가
- 안드로이드/IOS 배포



유지보수/ 사용자유치

- 모니터링과 AB테스트등 진행
- 버그추정
- 지속적인유지보수

향후 추가 기능

미래 적용 기능



미디어 반응으로 더 빠르고 재미있는 소통



롤링픽쳐: 릴레이 형식의 공유



큐레이션: 추억을 더 가치있게



인공 지능 기반의 사진 검색

미래 적용 기능



미디어 반응



빠른 반응을 통해 사용자의 상호작용을 유도하고 다양한 반응을 확인



:3indblown leaf add this to Oct 10



30min ago



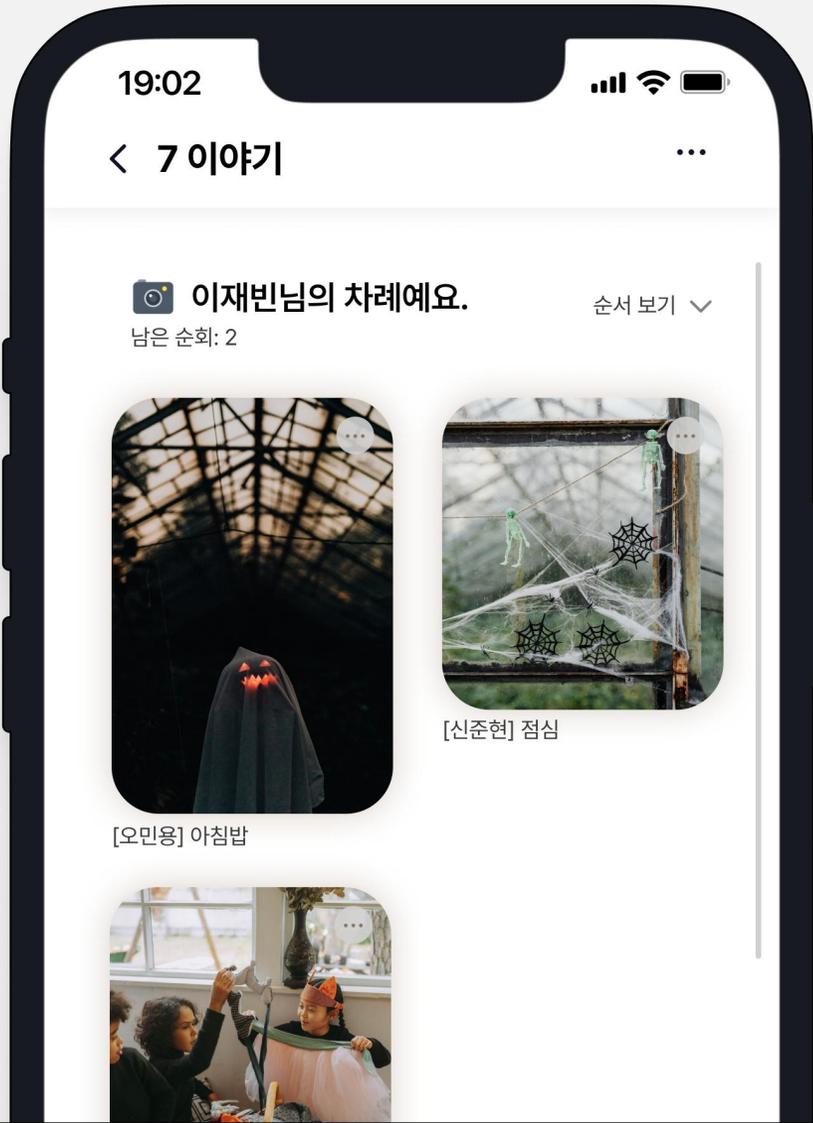
:3indblown leaf add this to Oct 10



30min ago



롤링 픽처



롤링 픽처

릴레이 형식의 앨범 저장법

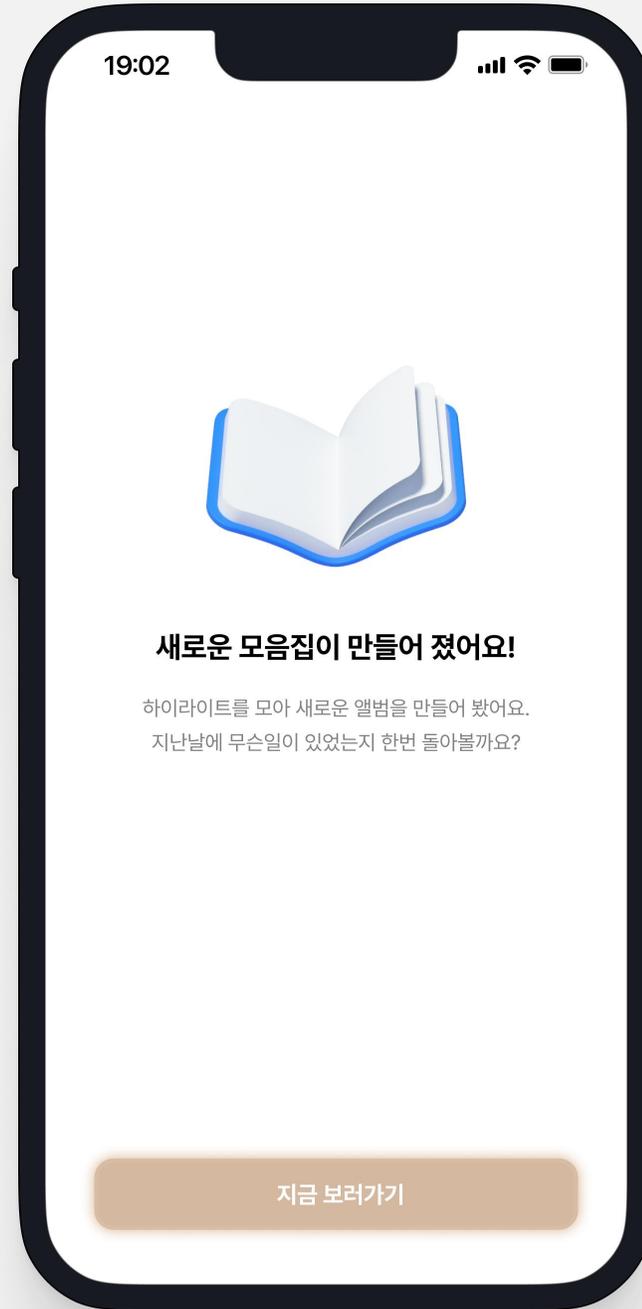
긴장감 있고 능동적인 업로드 장려

큐레이션

큐레이션

하이라이트만 모아
새로운 앨범으로 재탄생

앨범을 기반으로 하이라이트를 뽑아 새로운
앨범으로 만드는 큐레이션 기능을 추가



Smile DeveloperS



이재빈

백엔드, 디자인, 기획



오민용

앱, 기획, 노트 테이킹

고마워요



장영달M



이승윤M



한국디지털미디어고등학교

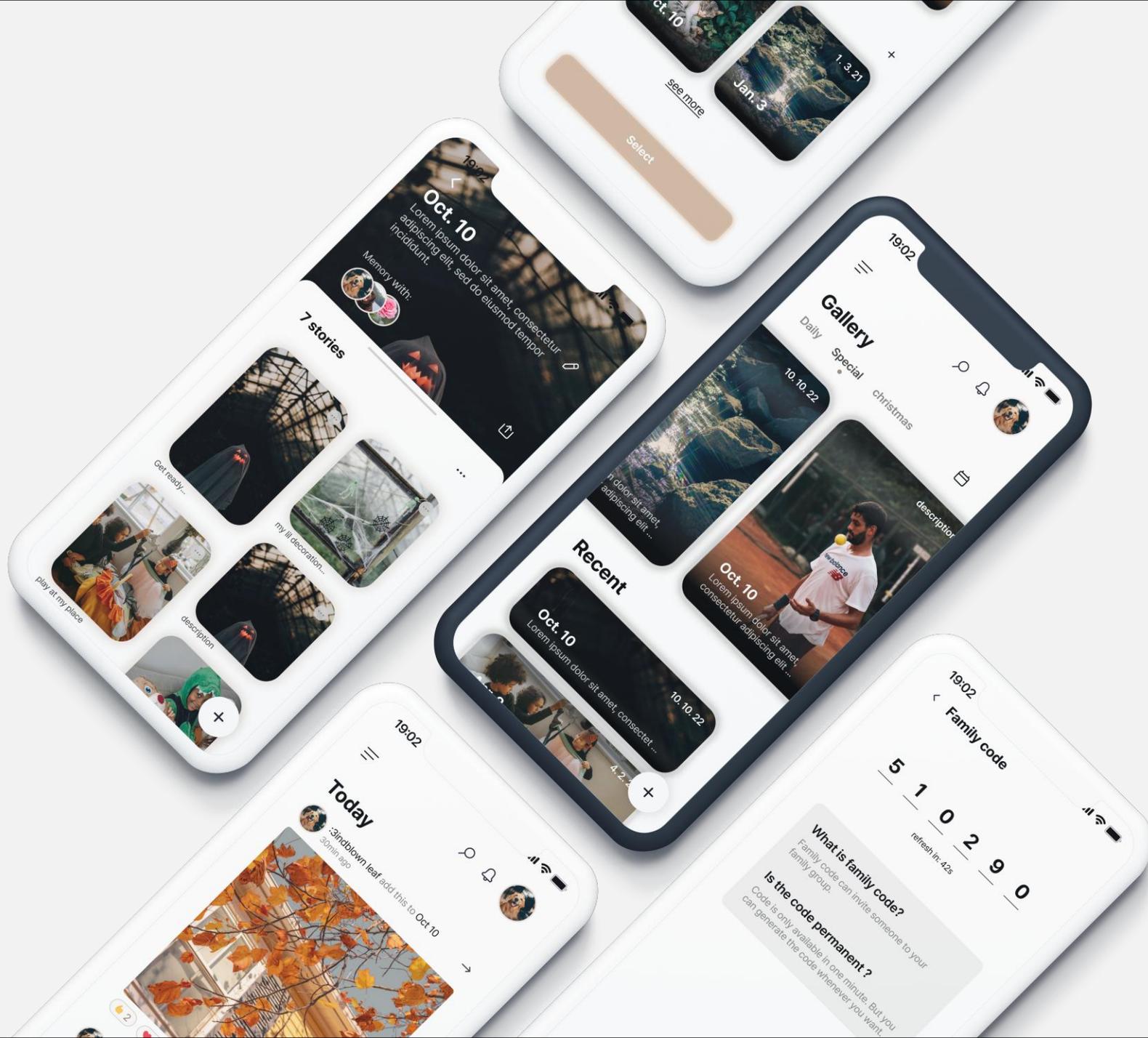


네이버클라우드플랫폼



파스타

기억을 기억하다, **Backup**



당신의 뒷 이야기를 기억해줄 Back U P

Smile Developers
이재빈 & 오민용

end of presentation