Unity Backend

“우편” 기능을 이용한 유저 우편 관리 (Console View)

Created in 2023-06-19

Last Updated 2023-06-19

Unity Version 2022.2.2f1

Index

- 재화 차트 생성 및 등록
- 콘솔에서 우편 발송
- 수령 가능한 우편 리스트 불러오기
- 우편 [하나/전체] 수령

재화 차트 생성 및 등록

- 재화 차트 생성
- 재화 차트 등록

재화 차트 생성 및 등록

재화 차트 생성

- 게임에 사용할 재화 데이터를 엑셀 파일을 이용해 생성

GoodsChart.xlsx - Excel

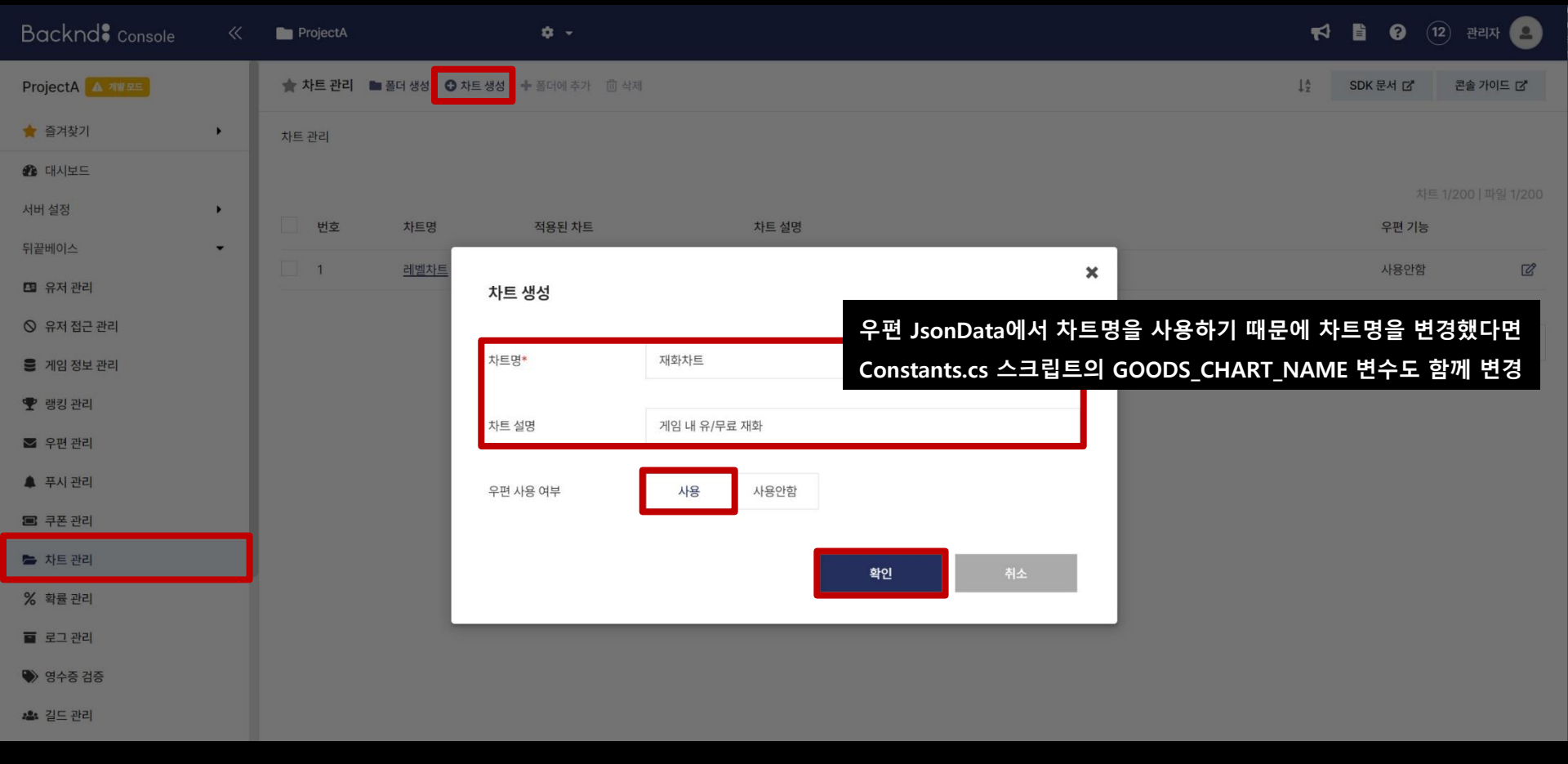
	A	B	C	D	E
1	itemId	itemName	itemInfo		
2	0	heart	게임을 플레이할 때 소모		
3	1	gold	게임 내 아이템을 구매하는데 사용하는 재화		
4	2	jewel	유료 아이템, 게임 내 재화 구매에 사용하는 재화		
5					
6			GoodsChart.xlsx		
7			게임을 플레이할 때 소모하는 heart, 게임 내 유료 재화 jewel,		
8			무료 재화 gold 정보		
9					
10					
11					
12					
13					

goodsChart

재화 차트 생성 및 등록

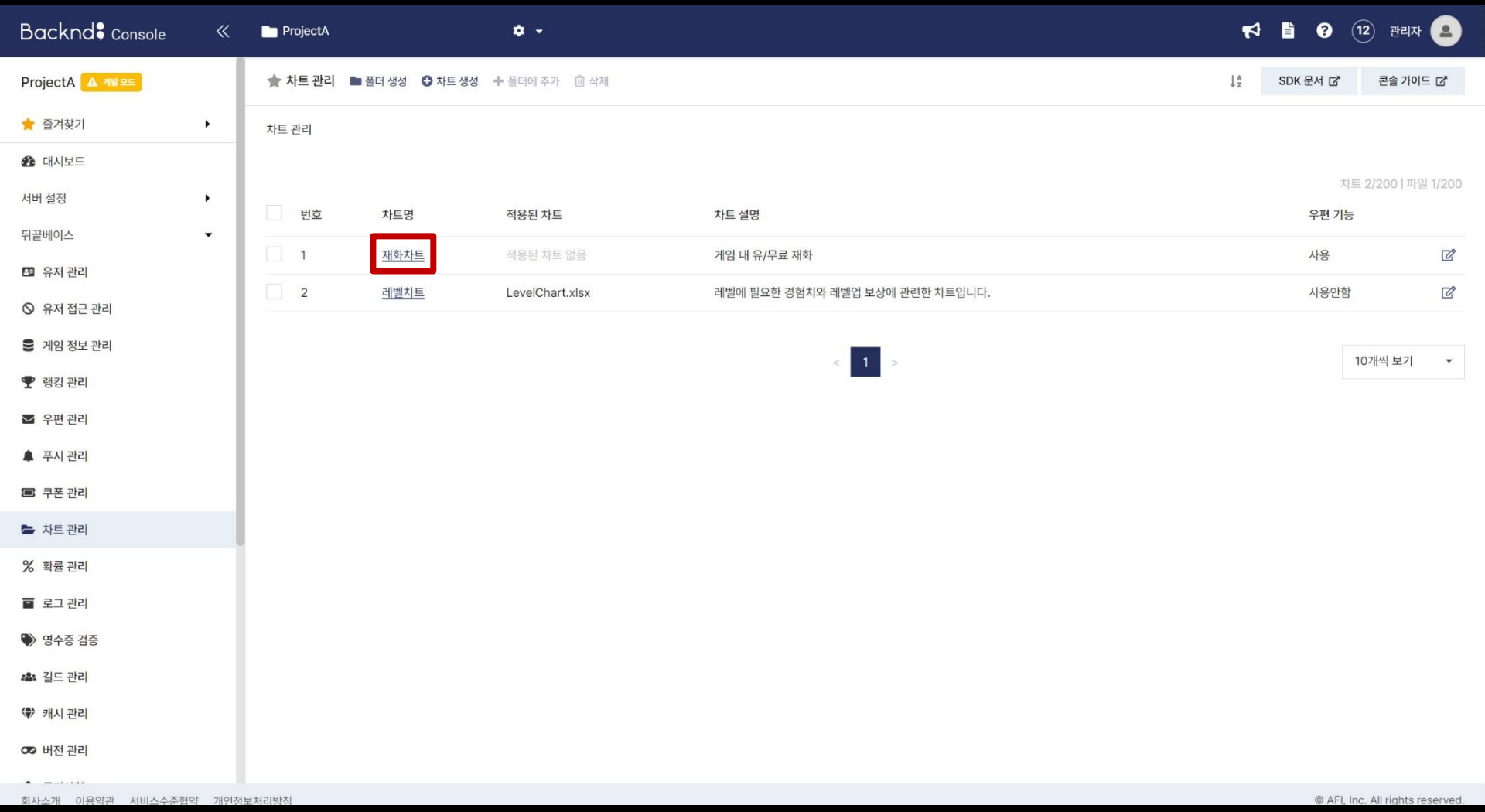
재화 차트 등록

- Backend Console의 "차트 관리" 탭에 차트 생성
  - 차트 관리 - "차트 생성"



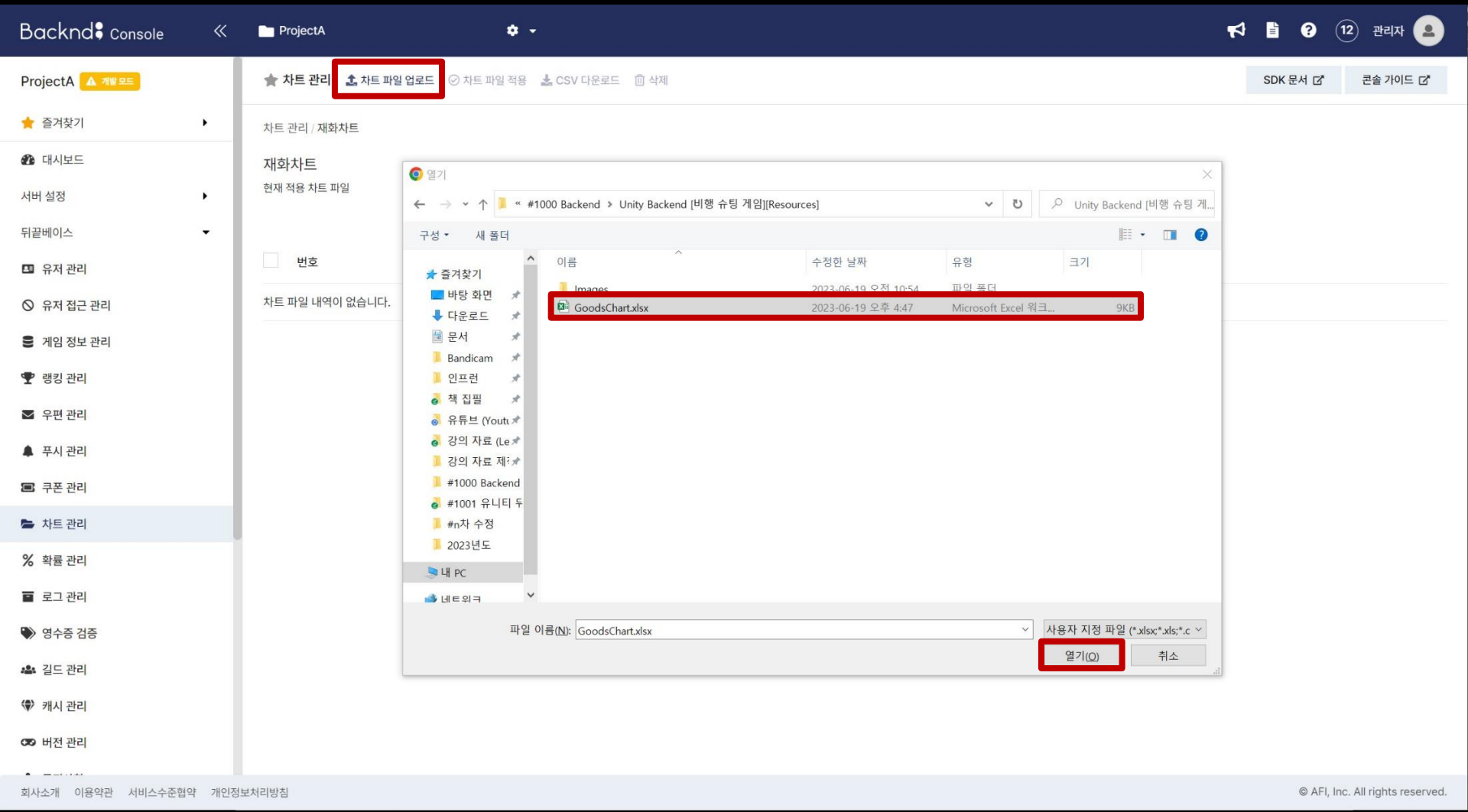
재화 차트 생성 및 등록

- 생성이 완료된 차트로 들어가 차트 파일 업로드
  - 차트 파일을 업로드할 차트 선택 : "재화차트" (차트명)



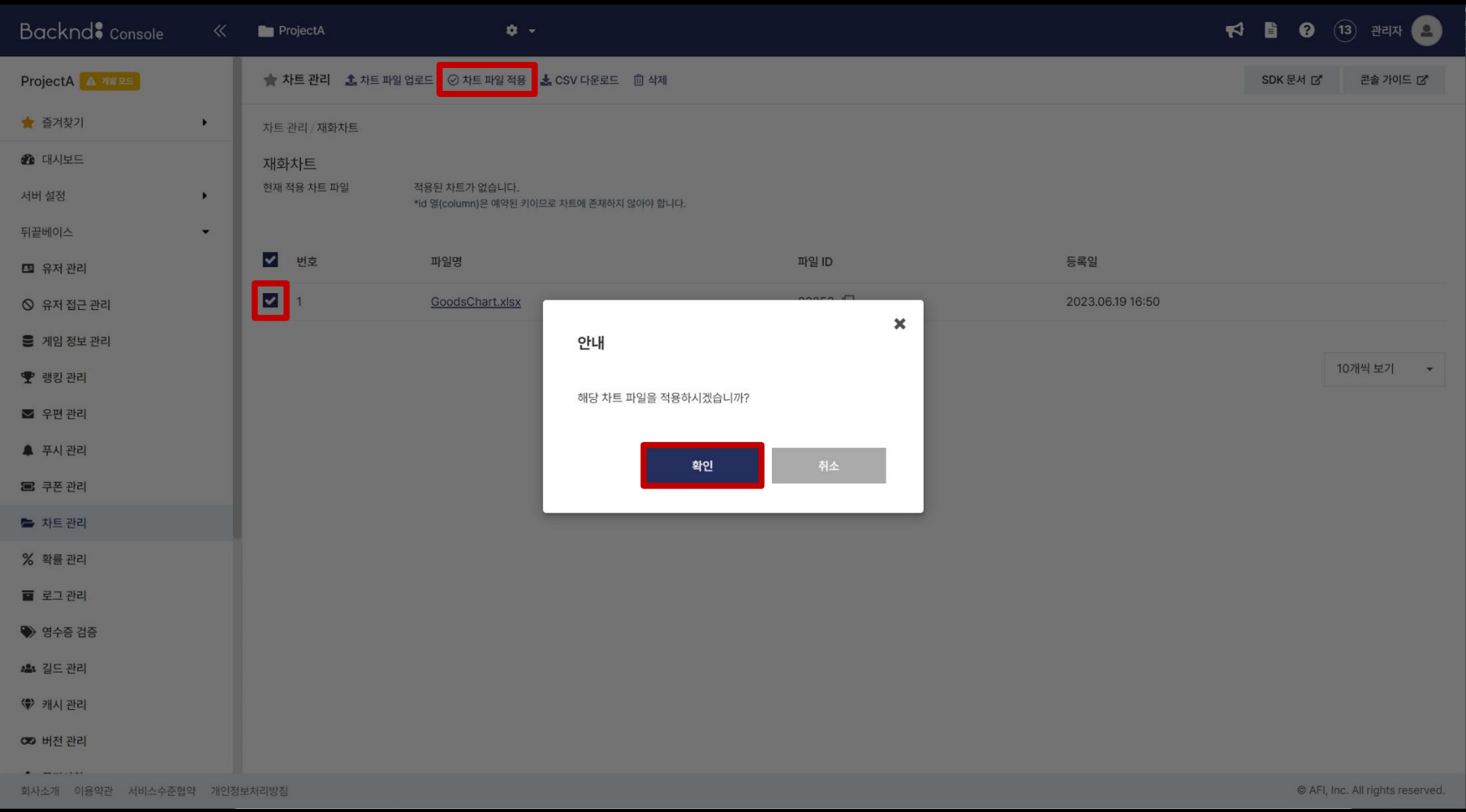
재화 차트 생성 및 등록

- "차트 파일 업로드" 선택 - GoodsChart.xlsx 업로드



재화 차트 생성 및 등록

- 차트 파일을 선택하고, "차트 파일 적용" 버튼을 눌러 차트 파일 적용



콘솔에서 우편 발송

- 우편 종류, 만료일, 삭제 시점
- 콘솔에서 우편 발송

콘솔에서 우편 발송

우편 종류, 만료일, 삭제 시점

우편 종류

구분	설명
관리자 우편	뒤끝 콘솔에서 발송하는 관리자 우편
쿠폰 우편	뒤끝 콘솔의 웹 쿠폰 설정에서 생성한 페이지로 쿠폰을 사용 후 발송되는 쿠폰 우편
랭킹 보상 우편	랭킹 보상 후 자동으로 지급되는 랭킹 우편
유저 우편	유저 간 게임정보 관리의 데이터를 이용해 발송한 유저 우편

콘솔에서 우편 발송

우편 만료일

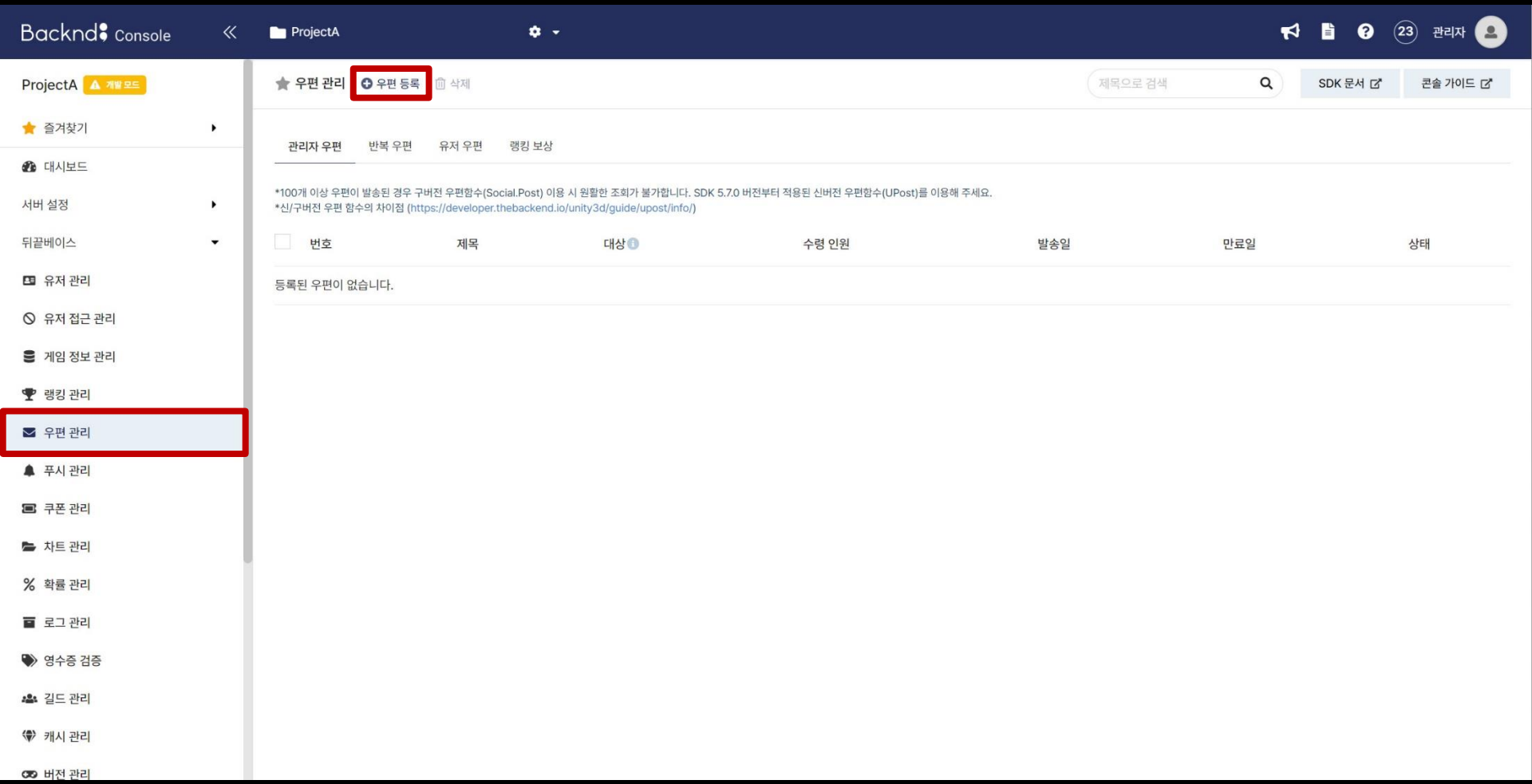
구분	만료일
관리자 우편	콘솔에서 설정한 시간 (1시간, 3시간, 6시간, 12시간, 24시간, 7일, 15일, 30일)
쿠폰 우편	쿠폰 수령 후 우편 발송 시 제한 없음 (쿠폰이 만료될 경우 우편 발송이 되지 않음)
랭킹 보상 우편	7일
유저 우편	15일

우편 삭제 시점

구분	삭제 시점
관리자 우편	유저가 우편 수령 시
쿠폰 우편	유저가 우편 수령 시
랭킹 보상 우편	유저가 우편 수령 시
유저 우편	우편 삭제 함수 호출 시 (재수령 불가)

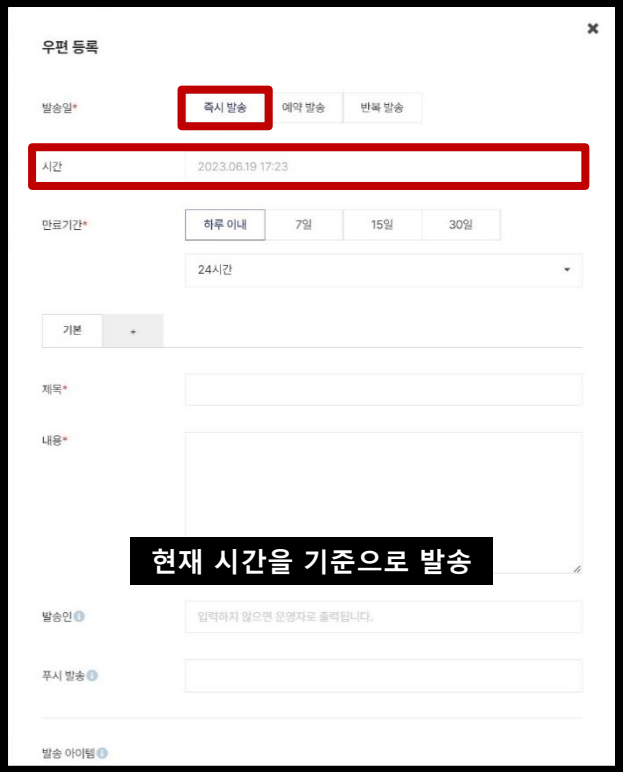
콘솔에서 우편 발송

- 콘솔에서 우편 발송
  - 우편 등록

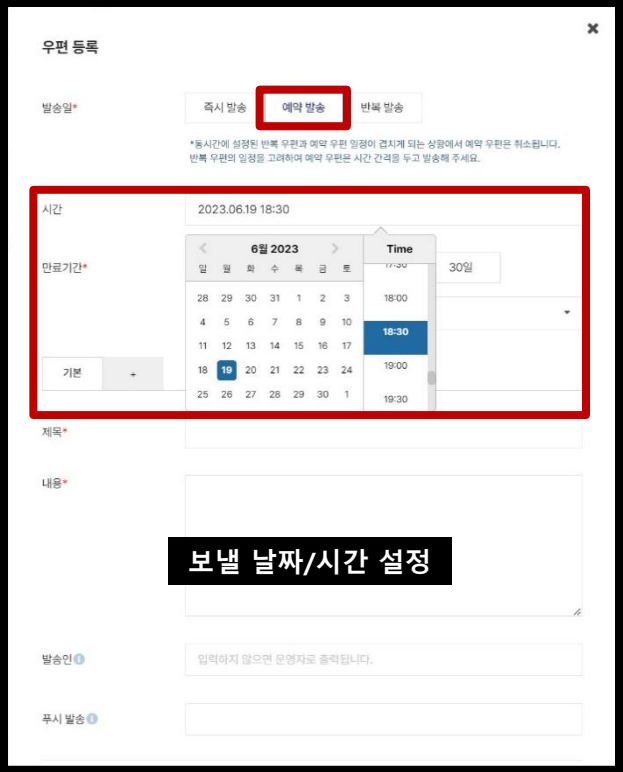


콘솔에서 우편 발송

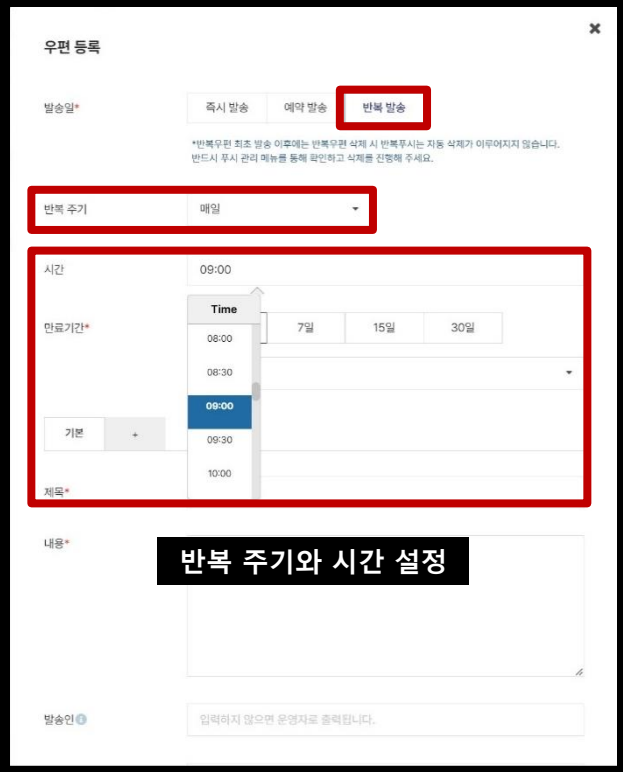
- 우편 발송일



< 즉시 발송 >



< 예약 발송 >



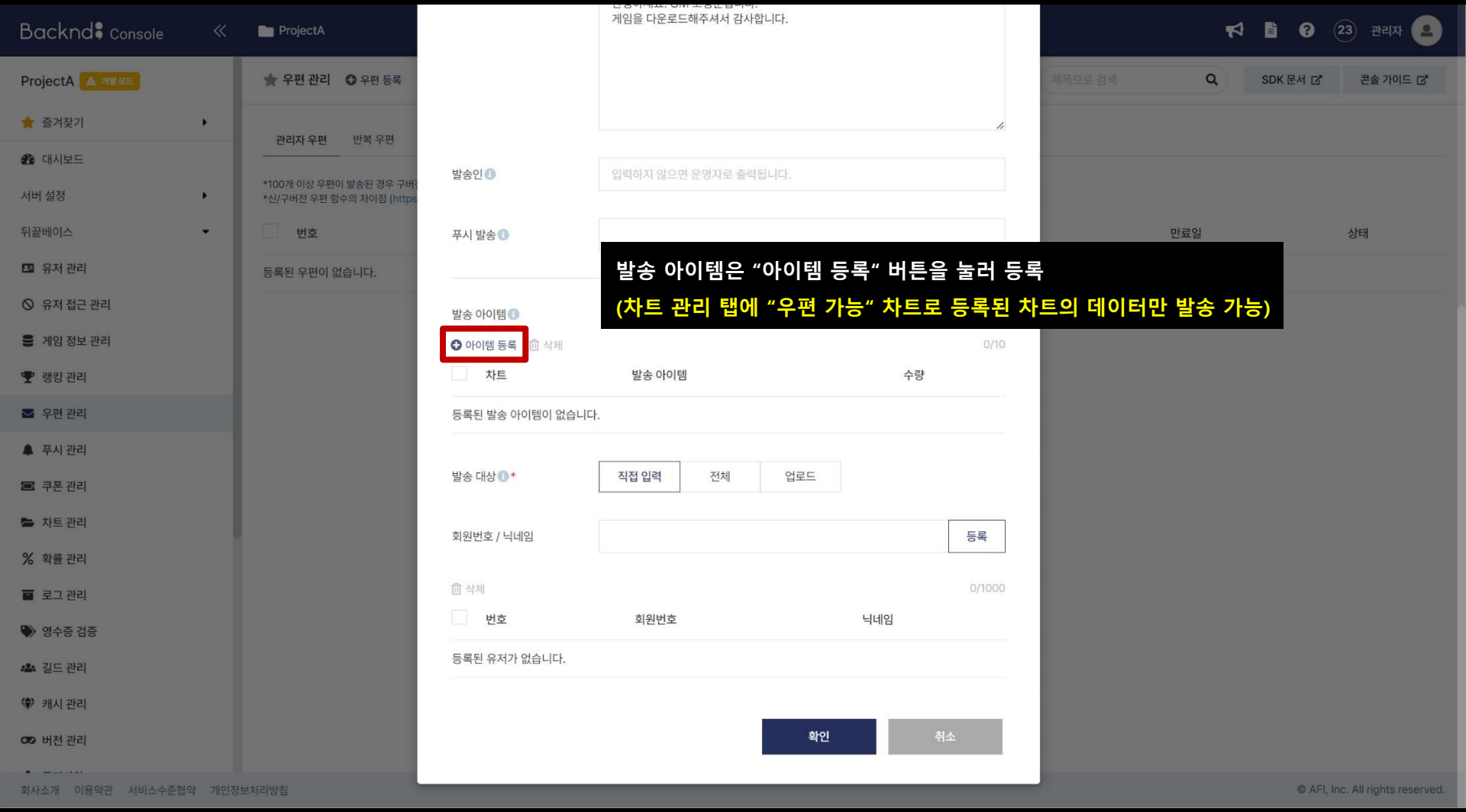
< 반복 발송 >

콘솔에서 우편 발송

- 우편 정보 설정

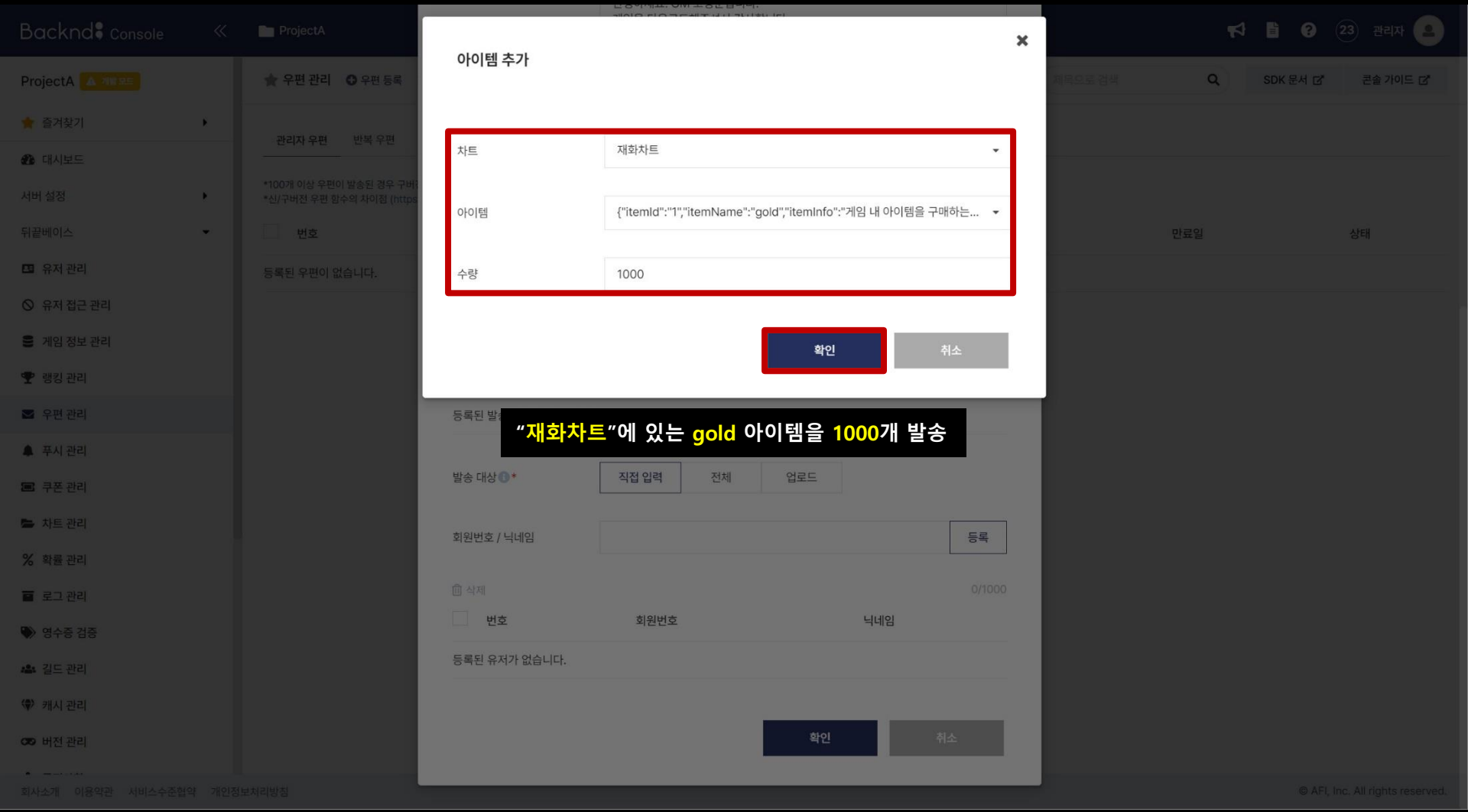
콘솔에서 우편 발송

- 우편 정보 설정 (계속)



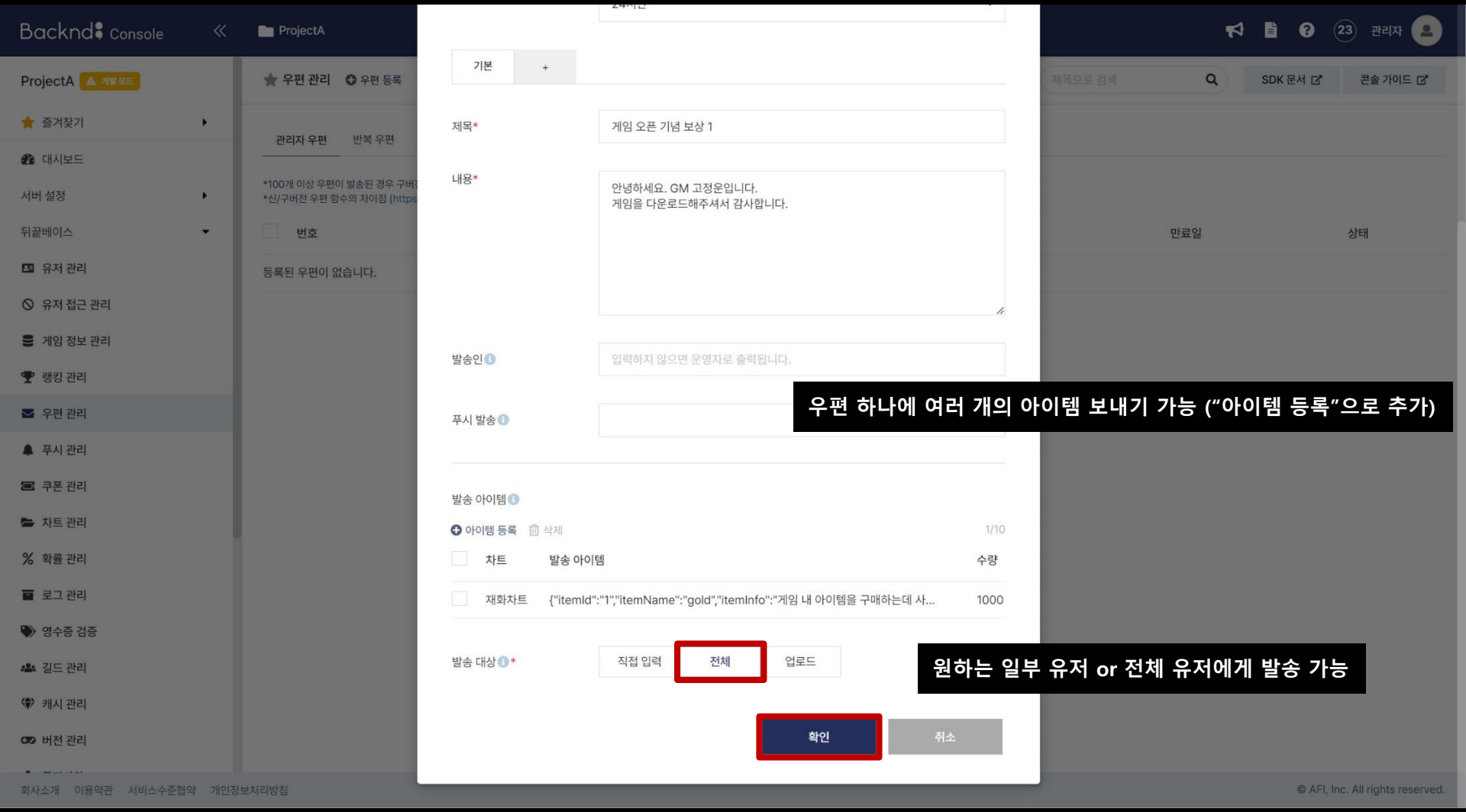
콘솔에서 우편 발송

- 우편 정보 설정 (계속)



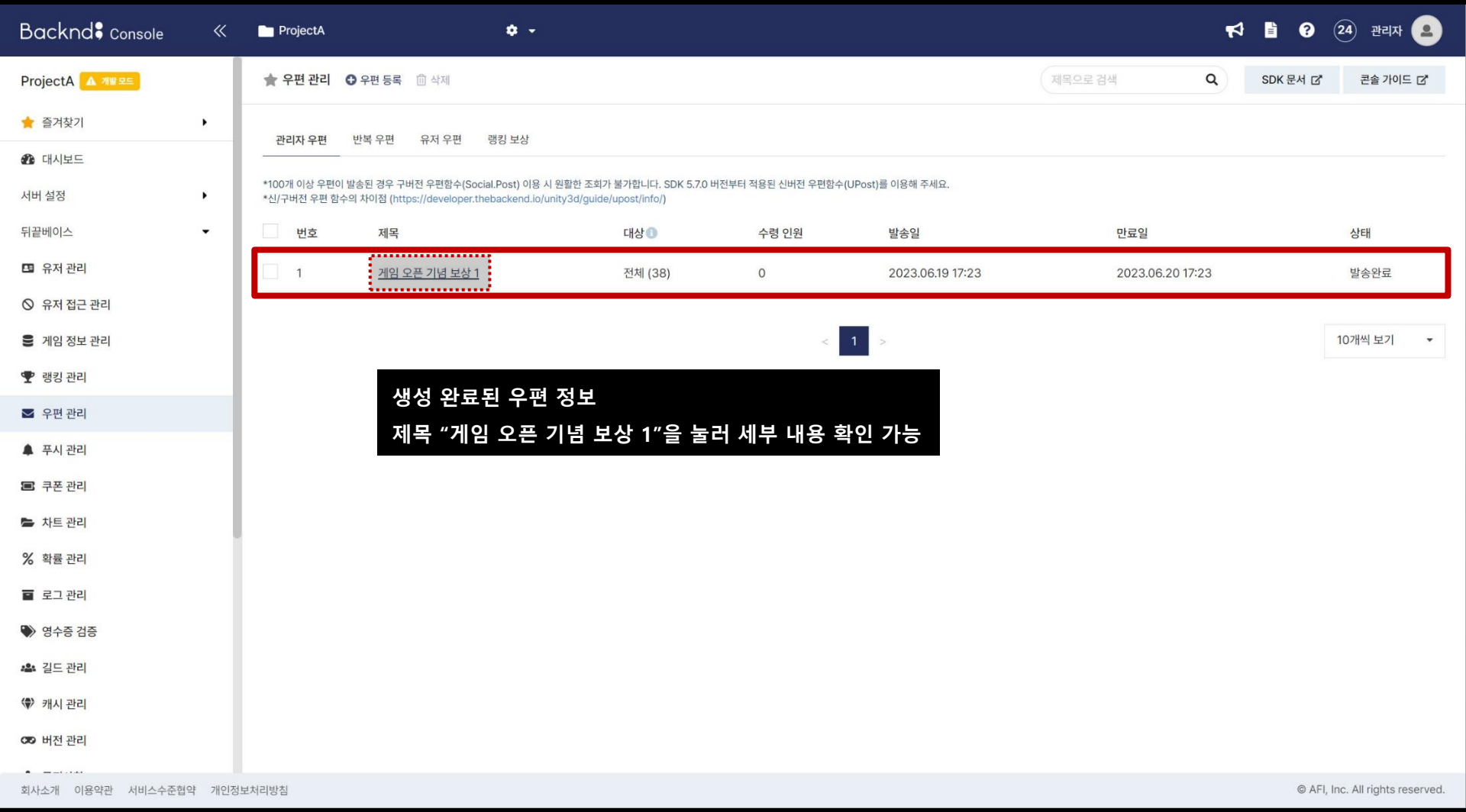
콘솔에서 우편 발송

- 우편 정보 설정 (계속)



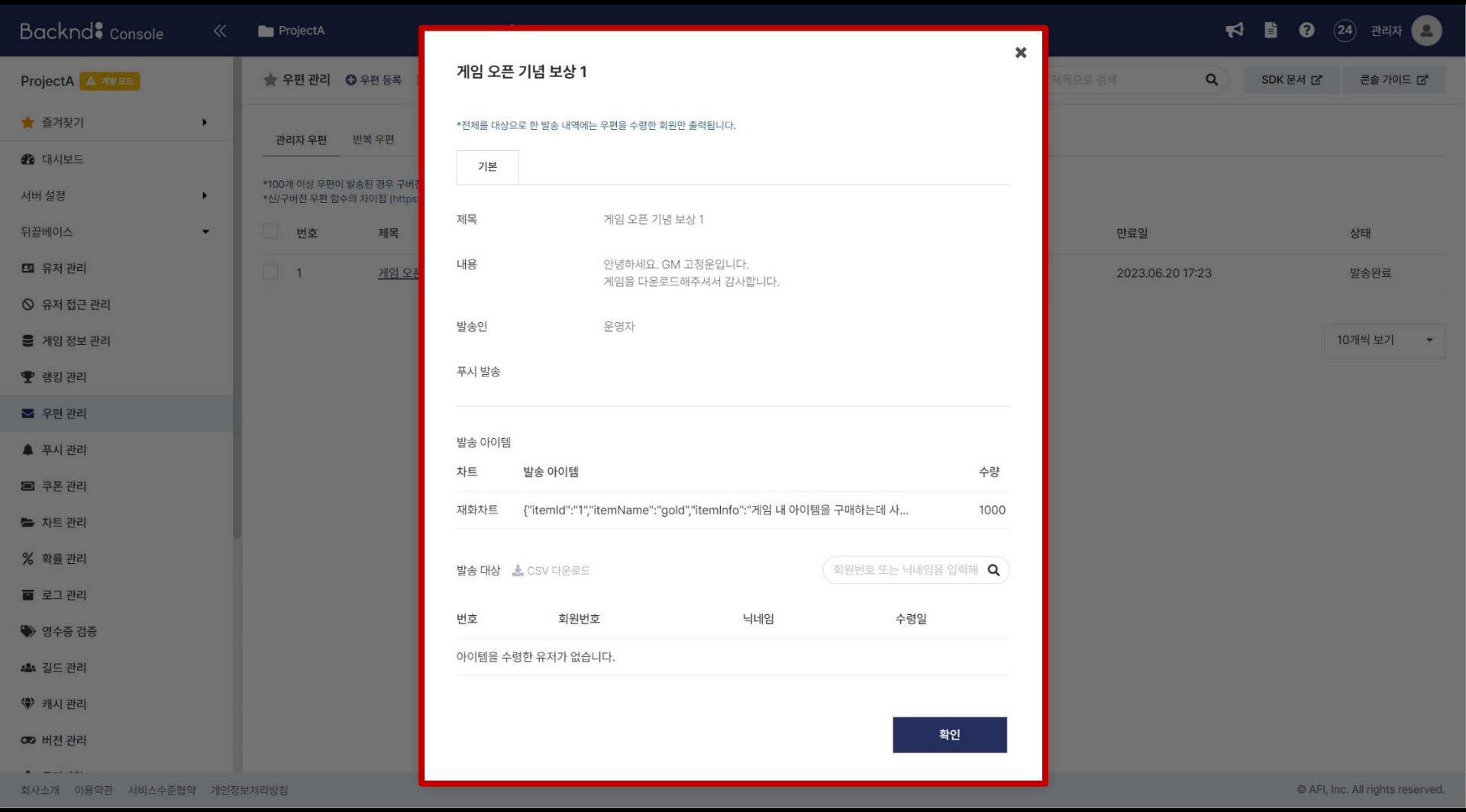
콘솔에서 우편 발송

- 우편 등록 완료



콘솔에서 우편 발송

- 우편 등록 완료 (계속)



콘솔에서 우편 발송

- 우편 추가 등록

Backnd Console

ProjectA

25 관리자

ProjectA 개발 모드

즐겨찾기

대시보드

서버 설정

뒤끝베이스

유저 관리

유저 접근 관리

게임 정보 관리

랭킹 관리

우편 관리

푸시 관리

쿠폰 관리

차트 관리

확률 관리

로그 관리

영수증 검증

길드 관리

캐시 관리

버전 관리

우편 관리 | 우편 등록 | 삭제

제목으로 검색

SDK 문서 | 콘솔 가이드

관리자 우편 | 반복 우편 | 유저 우편 | 랭킹 보상

*100개 이상 우편이 발송된 경우 구버전 우편함수(Social.Post) 이용 시 원활한 조회가 불가합니다. SDK 5.7.0 버전부터 적용된 신버전 우편함수(UPost)를 이용해 주세요.
*신/구버전 우편 함수의 차이점 (https://developer.thebackend.io/unity3d/guide/upost/info/)

번호	제목	대상	수령 인원	발송일	만료일	상태
1	게임 오픈 기념 보상 2	전체 (38)	0	2023.06.19 18:42	2023.06.20 18:42	발송완료
2	게임 오픈 기념 보상 1	전체 (38)	0	2023.06.19 17:23	2023.06.20 17:23	발송완료

제목 : 게임 오픈 기념 보상 2
아이템 : gold / 500개
나머지 내용 동일

< 1 >

10개씩 보기

회사소개 이용약관 서비스수준협약 개인정보처리방침

수령 가능한 우편 리스트 불러오기

- 우편 리스트 불러오기 메소드와 JsonData
- 우편 리스트 불러오기

수령 가능한 우편 리스트 불러오기

우편 리스트 불러오기 메소드와 JsonData

우편 리스트 불러오기 메소드 GetPostList()

```
GetPostList(PostType postType, int limit);
postType 속성의 우편을 limit 개수만큼 불러온다.
```

파라미터

구분	설명
postType	불러올 우편의 종류
limit	불러올 우편의 개수로 기본 값은 10 (10 <= limit <= 100)

PostType

구분	설명
Admin	콘솔에서 발송하는 관리자 우편
Rank	랭킹 결산 후 자동으로 지급되는 랭킹 우편
Coupon	뒤끝 콘솔의 웹 쿠폰 설정에서 생성한 페이지로 쿠폰을 사용 후 발송되는 쿠폰 우편
User	유저끼리 자신의 데이터를 이용해 발송한 유저 우편

수령 가능한 우편 리스트 불러오기

- 우편을 불러올 때 JsonData (관리자 우편 기준)

Tip. 우편 속성(PostType)에 따라 JsonData 일부 목록이 다름
[뒤끝 SDK - 우편 기능 - 우편 리스트 불러오기]

```
{
    "postList": [
        {
            "content": "우편 내용",                          // 콘솔에서 보낸 우편의 내용
            "expirationDate": "2023-06-19T02:22:22.022Z",    // 우편이 만료되는 날짜
            "reservationDate": "2023-06-19T02:22:22.022Z",   // 우편을 등록한 날짜
            "nickname": “noname",                            // 보내는 이의 닉네임
            "inDate": "2023-06-19T02:22:22.022Z",            // 우편의 inDate
            "title": "오픈 기념 보상”,                        // 우편의 제목
            "author": “GM고박사”,                             // 우편의 발송인
            "sentDate": "2023-06-19T02:22:22.022Z",          // 보낸 날짜
            "items": []                                      // 우편에 첨부된 아이템
        },
    ]
}
```

수령 가능한 우편 리스트 불러오기

- 우편을 불러올 때 JsonData (관리자 우편 기준) (계속)

```
{
    "postList": [
        {
            ..
            "items": [
            {
                "item": {
                    "chartFileName": "GoodsChart.xlsx"          // 차트 파일 이름(*.xlsx)
                    "itemId": "01"
                    "itemName": "gold"
                    "itemInfo": "게임 내 아이템을 구매하는데 사용하는 재화"
                }
                "itemCount": 1000,                              // 아이템의 개수
                "chartName": "재화차트"                          // 차트 이름
            }
        ]
        },
    ]
}
```

A	B	C
itemId	itemName	itemInfo
0	heart	게임을 플레이할 때 소모
1	gold	게임 내 아이템을 구매하는데 사용하는 재화
2	jewel	유료 아이템, 게임 내 재화 구매에 사용하는 재화

발송 아이템

차트	발송 아이템	수량
재화차트	{"itemId":"1","itemName":"gold","itemInfo":"게임 내 아이템을 구매하는데 사...	1000

Tip. "item" 내부의 정보는 등록한 차트의 행 제목에 따라 달라진다.

수령 가능한 우편 리스트 불러오기

- 우편 리스트 불러오기
  - 재화 차트 이름 변수 선언 및 설정
    - Constants Script 수정

```
1   public static class Constants
2   {
3       public static readonly string   USER_DATA_TABLE     = "USER_DATA";
4       public static readonly string   DAILY_RANK_UUID     = "0fa14490-085a-11ee-b506-7d5eaba96eff";
5       public static readonly int      MAX_RANK_LIST       = 20;
6
7       public static readonly string   LEVEL_CHART         = "82683";
8
9       public static readonly string   GOODS_CHART_NAME    = "재화차트";
10  }
```

수령 가능한 우편 리스트 불러오기

- 우편 하나의 데이터를 관리하는 스크립트 생성 및 작성
  - C# Script 생성 후 스크립트의 이름을 "PostData"로 변경

```
1   using System.Collections.Generic;
2
3   public class PostData
4   {
5       public  string  title;                  // 우편 제목
6       public  string  content;                // 우편 내용
7       public  string  inDate;                 // 우편 inDate
8       public  string  expirationDate;         // 우편 만료 날짜
9
10      public  bool    isCanReceive = false;   // 우편에 받을 수 있는 아이템이 있는지 여부
11
12      // <아이템 이름, 아이템 개수>
13      public  Dictionary<string, int> postReward = new Dictionary<string, int>();
14
```

수령 가능한 우편 리스트 불러오기

- 우편 하나의 데이터를 관리하는 스크립트 생성 및 작성 (계속)

```
15      // 우편 정보를 Debug.Log()로 출력하기 위한 메소드
16      public override string ToString()
17      {
18          string result = string.Empty;
19          result += $"title : {title}\n";
20          result += $"content : {content}\n";
21          result += $"inDate : {inDate}\n";
22
23          if ( isCanReceive )
24          {
25              result += "우편 아이템\n";
26
27              foreach ( string itemKey in postReward.Keys )
28              {
29                  result += $"| {itemKey} : {postReward[itemKey]}개\n";
30              }
31          }
32          else
33          {
34              result += "지원하지 않는 아이템입니다.";
35          }
36
37          return result;
38      }
39  }
```

수령 가능한 우편 리스트 불러오기

- 수령 가능한 우편 리스트를 불러오는 스크립트 생성 및 작성
  - C# Script 생성 후 스크립트의 이름을 "BackendPostSystem"으로 변경

```
1   using System.Collections.Generic;
2   using UnityEngine;
3   using BackEnd;
4
5   public class BackendPostSystem : MonoBehaviour
6   {
7       private List<PostData> postList = new List<PostData>();
8
9       private void Update()
10      {
11          if ( Input.GetKeyDown("1") )
12          {
13              PostListGet(PostType.Admin);
14          }
15      }
16
```

수령 가능한 우편 리스트 불러오기

- 수령 가능한 우편 리스트를 불러오는 스크립트 생성 및 작성 (계속)

```
17      public void PostListGet(PostType postType)
18      {
19          Backend.UPost.GetPostList(postType, callback =>
20          {
21              if ( !callback.IsSuccess() )
22              {
23                  Debug.LogError($"우편 불러오기 중 에러가 발생했습니다 : {callback}");
24                  return;
25              }
26
27              Debug.Log($"우편 리스트 불러오기 요청에 성공했습니다 : {callback}");
28
29              // JSON 데이터 파싱 성공
30              try
31              {
32                  LitJson.JsonData jsonData = callback.GetFlattenJSON()["postList"];
33
34                  // 받아온 데이터의 개수가 0이면 데이터가 없는 것
35                  if ( jsonData.Count <= 0 )
36                  {
37                      Debug.LogWarning("우편함이 비어있습니다.");
38                      return;
39                  }
40
41                  // 매번 우편 리스트를 불러올 때 postList 초기화
42                  postList.Clear();
43
```

수령 가능한 우편 리스트 불러오기

- 수령 가능한 우편 리스트를 불러오는 스크립트 생성 및 작성 (계속)

```
44              // 현재 저장 가능한 모든 우편 정보 불러오기
45              for ( int i = 0; i < jsonData.Count; ++ i )
46              {
47                  PostData post       = new PostData();
48
49                  post.title          = jsonData[i]["title"].ToString();
50                  post.content        = jsonData[i]["content"].ToString();
51                  post.inDate         = jsonData[i]["inDate"].ToString();
52                  post.expirationDate = jsonData[i]["expirationDate"].ToString();
53
54                  // 우편에 함께 발송된 모든 아이템 정보 불러오기
55                  foreach ( LitJson.JsonData itemJson in jsonData[i]["items"] )
56                  {
57                      // 우편에 함께 발송된 아이템의 차트 이름이 "재화차트" 일 때
58                      if ( itemJson["chartName"].ToString() == Constants.GOODS_CHART_NAME )
59                      {
60                          string  itemName    = itemJson["item"]["itemName"].ToString();
61                          int     itemCount   = int.Parse(itemJson["itemCount"].ToString());
62
63                          // 우편에 포함된 아이템이 여러 개 일 때
64                          // 이미 postReward에 해당 아이템 정보가 있으면 개수 추가
65                          if ( post.postReward.ContainsKey(itemName) )
66                          {
67                              post.postReward[itemName] += itemCount;
68                          }
69                          // postReward에 없는 아이템 정보이면 요소 추가
70                          else
71                          {
72                              post.postReward.Add(itemName, itemCount);
73                          }
74
75                          post.isCanReceive = true;
76                      }
```

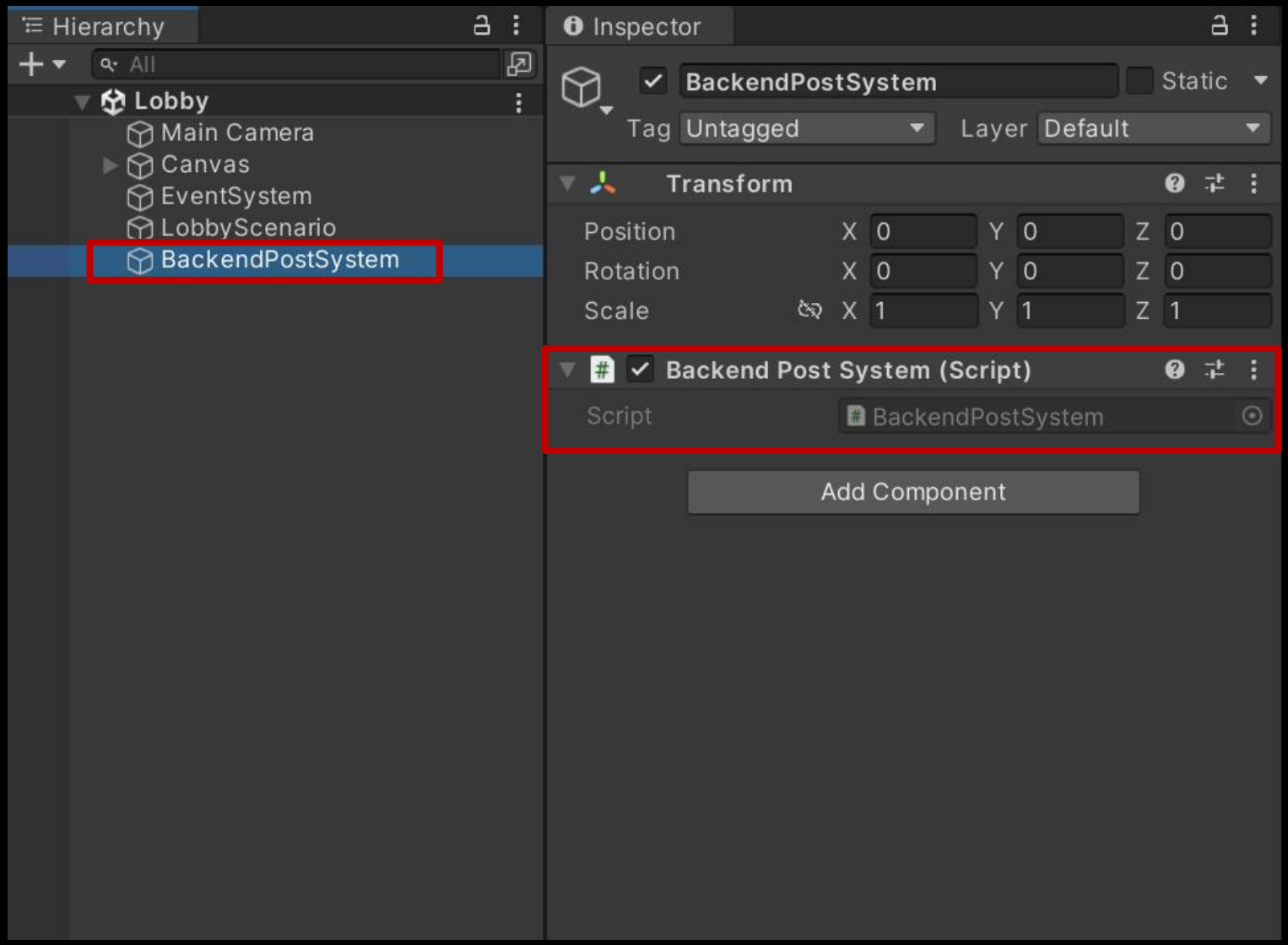
수령 가능한 우편 리스트 불러오기

- 수령 가능한 우편 리스트를 불러오는 스크립트 생성 및 작성 (계속)

```
77                          else
78                          {
79                              Debug.LogWarning($"아직 지원되지 않는 차트 정보입니다. : {itemJson["chartName"].ToString()}");
80
81                              post.isCanReceive = false;
82                          }
83                      }
84
85                      postList.Add(post);
86                  }
87
88                  // 저장 가능한 모든 우편(postList) 정보 출력
89                  for ( int i = 0; i < postList.Count; ++ i )
90                  {
91                      Debug.Log($"{i}번째 우편\n{postList[i].ToString()}");
92                  }
93              }
94              // JSON 데이터 파싱 실패
95              catch ( System.Exception e )
96              {
97                  // try-catch 에러 출력
98                  Debug.LogError(e);
99              }
100         });
101     }
102 }
```

수령 가능한 우편 리스트 불러오기

- BackendPostSystem 오브젝트에 "BackendPostSystem" 컴포넌트 추가



수령 가능한 우편 리스트 불러오기

- 결과 화면 (Unity Editor)

Console

Clear | Collapse | Error Pause | Editor

[18:44:46] Level : 3, Max Exp : 400,Reward Gold : 3333
UnityEngine.Debug:Log (object)

[18:44:46] Level : 4, Max Exp : 800,Reward Gold : 4444
UnityEngine.Debug:Log (object)

[18:44:46] Level : 5, Max Exp : 1600,Reward Gold : 5555
UnityEngine.Debug:Log (object)

[18:44:46] Level : 6, Max Exp : 3200,Reward Gold : 6666
UnityEngine.Debug:Log (object)

[18:44:46] Level : 7, Max Exp : 6400,Reward Gold : 7777
UnityEngine.Debug:Log (object)

[18:44:46] Level : 8, Max Exp : 12800,Reward Gold : 8888
UnityEngine.Debug:Log (object)

[18:44:46] Level : 9, Max Exp : 25600,Reward Gold : 9999
UnityEngine.Debug:Log (object)

[18:44:46] Level : 10, Max Exp : 51200,Reward Gold : 100000
UnityEngine.Debug:Log (object)

[18:44:46] 게임 정보 데이터 불러오기에 성공했습니다. : statusCode : 200
message : Success

[18:44:49] 우편 리스트 불러오기 요청에 성공했습니다 : statusCode : 200
message : Success

[18:44:49] 0번째 우편
title : 게임 오픈 기념 보상 2

[18:44:49] 1번째 우편
title : 게임 오픈 기념 보상 1

0번째 우편
title : 게임 오픈 기념 보상 2
content : 안녕하세요. GM 고정운입니다.
게임을 다운로드해주셔서 감사합니다.
inDate : 2023-06-19T09:42:06.057Z
우편 아이템
| gold : 500개

GetPostList()로 불러온 우편은 최신 순으로 정렬된다.

Console

Clear | Collapse | Error Pause | Editor

[18:44:46] Level : 3, Max Exp : 400,Reward Gold : 3333
UnityEngine.Debug:Log (object)

[18:44:46] Level : 4, Max Exp : 800,Reward Gold : 4444
UnityEngine.Debug:Log (object)

[18:44:46] Level : 5, Max Exp : 1600,Reward Gold : 5555
UnityEngine.Debug:Log (object)

[18:44:46] Level : 6, Max Exp : 3200,Reward Gold : 6666
UnityEngine.Debug:Log (object)

[18:44:46] Level : 7, Max Exp : 6400,Reward Gold : 7777
UnityEngine.Debug:Log (object)

[18:44:46] Level : 8, Max Exp : 12800,Reward Gold : 8888
UnityEngine.Debug:Log (object)

[18:44:46] Level : 9, Max Exp : 25600,Reward Gold : 9999
UnityEngine.Debug:Log (object)

[18:44:46] Level : 10, Max Exp : 51200,Reward Gold : 100000
UnityEngine.Debug:Log (object)

터 불러오기에 성공했습니다. : statusCode : 200

[18:44:49] 우편 리스트 불러오기 요청에 성공했습니다 : statusCode : 200
message : Success

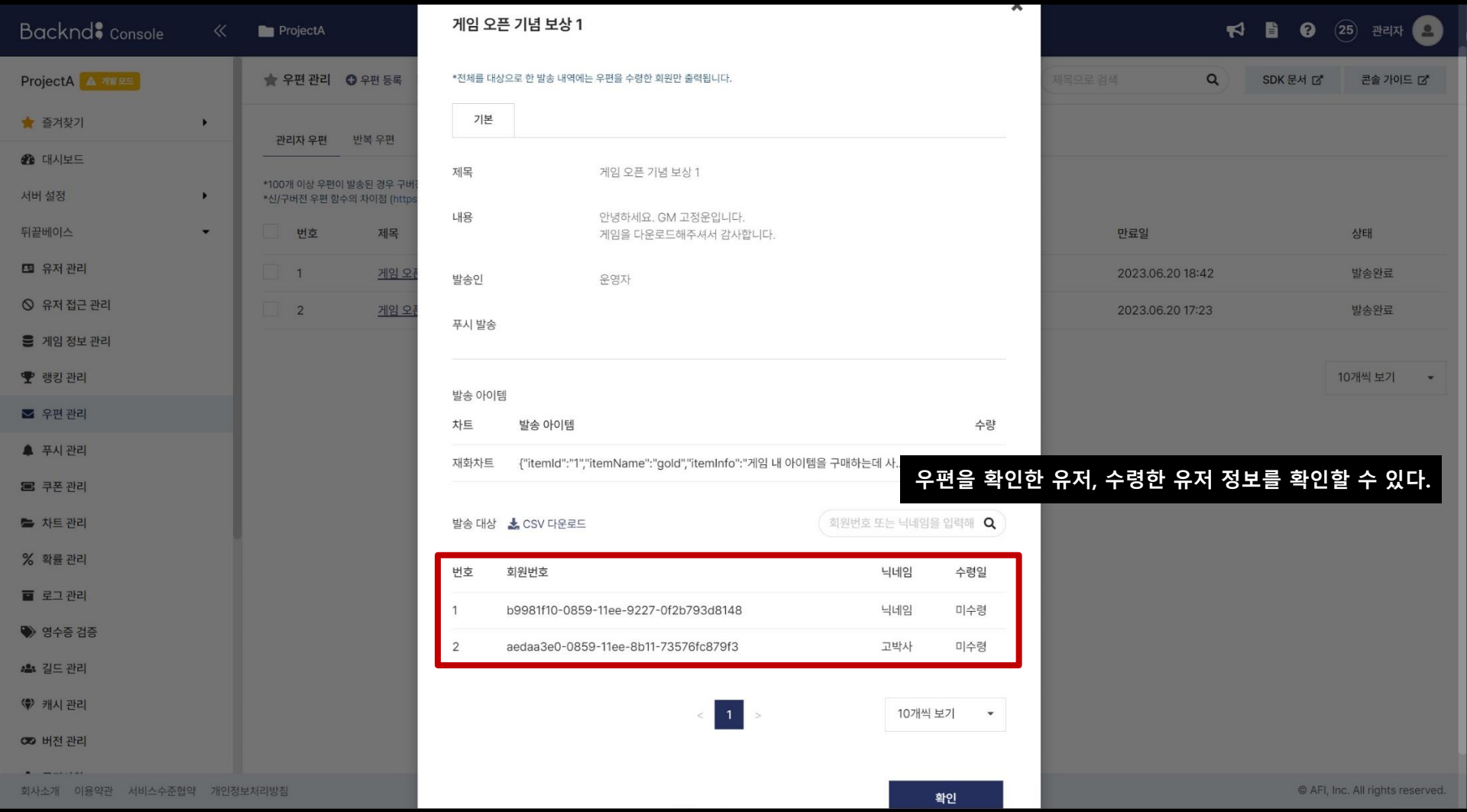
[18:44:49] 0번째 우편
title : 게임 오픈 기념 보상 2

[18:44:49] 1번째 우편
title : 게임 오픈 기념 보상 1

1번째 우편
title : 게임 오픈 기념 보상 1
content : 안녕하세요. GM 고정운입니다.
게임을 다운로드해주셔서 감사합니다.
inDate : 2023-06-19T08:23:33.835Z
우편 아이템
| gold : 1000개

수령 가능한 우편 리스트 불러오기

- 결과 화면 (Backend Console)



우편 [하나/전체] 수령

- 우편 하나 수령
- 우편 전체 수령

우편 [하나/전체] 수령

우편 하나 수령

- ReceivePostItem() 메소드를 이용한 index번째 우편 수령
  - BackendPostSystem Script 수정

```
1   using System.Collections.Generic;
2   using UnityEngine;
3   using BackEnd;
4
5   public class BackendPostSystem : MonoBehaviour
6   {
7       private List<PostData> postList = new List<PostData>();
8
9       private void Update()
10      {
11          if ( Input.GetKeyDown("1") )
12          {
13              PostListGet(PostType.Admin);
14          }
15          else if ( Input.GetKeyDown("2") )
16          {
17              PostReceive(PostType.Admin, 0);
18          }
19      }
20
21      public void PostListGet(PostType postType)...
106
```

우편 [하나/전체] 수령

- BackendPostSystem Script 수정 (계속)

```
107  public void PostReceive(PostType postType, int index)
108  {
109      if ( postList.Count <= 0 )
110      {
111          Debug.LogWarning("받을 수 있는 우편이 존재하지 않습니다. 혹은 우편 리스트 불러오기를 먼저 호출해주세요.");
112          return;
113      }
114
115      if ( index >= postList.Count )
116      {
117          Debug.LogError($"해당 우편은 존재하지 않습니다. : 요청 index{index} / 우편 최대 갯수 : {postList.Count}");
118          return;
119      }
120
121      Debug.Log($"{postType.ToString()}의 {postList[index].inDate} 우편수령을 요청합니다.");
122
```

우편 [하나/전체] 수령

- BackendPostSystem Script 수정 (계속)

```
123     Backend.UPost.ReceivePostItem(postType, postList[index].inDate, callback =>
124     {
125         if ( !callback.IsSuccess() )
126         {
127             Debug.LogError($"{postType.ToString()}의 {postList[index].inDate} 우편수령 중 에러가 발생했습니다. : {callback}");
128             return;
129         }
130 
131         Debug.Log($"{postType.ToString()}의 {postList[index].inDate} 우편수령에 성공했습니다. : {callback}");
132 
133         postList.RemoveAt(index);
134 
135         // 저장 가능한 아이템이 있을 때
136         if ( callback.GetFlattenJSON()["postItems"].Count > 0 )
137         {
138             // 아이템 저장
139             SavePostToLocal(callback.GetFlattenJSON()["postItems"]);
140             // 플레이어의 재화 정보를 서버에 업데이트
141             BackendGameData.Instance.GameDataUpdate();
142         }
143         else
144         {
145             Debug.LogWarning("수령 가능한 우편 아이템이 존재하지 않습니다.");
146         }
147     });
148 }
149 
```

Tip. 우편을 수령할 때 아이템의 키 값은 Items가 아닌 [“postItems”]

우편 [하나/전체] 수령

- 우편을 수령할 때 아이템의 JsonData

Tip. 우편을 수령할 때 아이템의 키 값은 Items가 아닌 [“postItems”]

```
{
    "postItems": [
        {
            “item”: {
                “chartFileName”: “GoodsChart.xlsx”
                “itemId”: “01”
                “itemName”: “gold”
                “itemInfo”: “게임 내 아이템을 구매하는데 사용하는 재화”
            }
            “itemCount”: 1000,
            “chartName”: “재화차트”
        },
    ]
}
```

우편 [하나/전체] 수령

- BackendPostSystem Script 수정 (계속)

```
150     public void SavePostToLocal(LitJson.JsonData item)
151     {
152         // JSON 데이터 파싱 성공
153         try
154         {
155             foreach ( LitJson.JsonData itemJson in item )
156             {
157                 // 차트 파일 이름(*.xlsx)과 Backend Console에 등록한 차트 이름
158                 string  chartFileName   = itemJson["item"]["chartFileName"].ToString();
159                 string  chartName       = itemJson["chartName"].ToString();
160 
161                 // GoodsChart.xlsx에 등록한 첫번째 행 이름
162                 int     itemId          = int.Parse(itemJson["item"]["itemId"].ToString());
163                 string  itemName        = itemJson["item"]["itemName"].ToString();
164                 string  itemInfo        = itemJson["item"]["itemInfo"].ToString();
165 
166                 // 우편 발송할 때 작성하는 아이템 수량
167                 int     itemCount       = int.Parse(itemJson["itemCount"].ToString());
168 
```

우편 [하나/전체] 수령

- BackendPostSystem Script 수정 (계속)

```
169                 // 우편으로 받은 재화를 게임 내 데이터에 적용
170                 if ( chartName.Equals(Constants.GOODS_CHART_NAME) )
171                 {
172                     if ( itemName.Equals("heart") )
173                     {
174                         BackendGameData.Instance.UserGameData.heart += itemCount;
175                     }
176                     else if ( itemName.Equals("gold") )
177                     {
178                         BackendGameData.Instance.UserGameData.gold += itemCount;
179                     }
180                     else if ( itemName.Equals("jewel") )
181                     {
182                         BackendGameData.Instance.UserGameData.jewel += itemCount;
183                     }
184                 }
185
186                 Debug.Log($"{chartName} - {chartFileName}");
187                 Debug.Log($"[{itemId}] {itemName} : {itemInfo}, 획득 수량 : {itemCount}");
188                 Debug.Log($"아이템을 수령했습니다. : {itemName} - {itemCount}개");
189             }
190         }
191         // JSON 데이터 파싱 실패
192         catch ( System.Exception e )
193         {
194             // try-catch 에러 출력
195             Debug.LogError(e);
196         }
197     }
198 }
```

우편 [하나/전체] 수령

- GameDataUpdate()를 호출했을 때도 TopPanel UI 정보가 갱신되도록 설정
  - BackendGameData Script 수정

```
122     /// <summary> 뒤끝 콘솔 테이블에 있는 유저 데이터 갱신
125     public void GameDataUpdate(UnityAction action=null)
126     {
127         if ( userGameData == null )...
133
134         Param param = new Param()...;
143
144         // 게임 정보의 고유값(gameDataRowInDate)이 없으면 에러 메시지 출력
145         if ( string.IsNullOrEmpty(gameDataRowInDate) )...
149         // 게임 정보의 고유값이 있으면 테이블에 저장되어 있는 값 중 inDate 컬럼의 값과
150         // 소유하는 유저의 owner_inDate가 일치하는 row를 검색하여 수정하는 UpdateV2() 호출
151         else
152         {
153             Debug.Log($"{gameDataRowInDate}의 게임 정보 데이터 수정을 요청합니다.");
154
155             Backend.GameData.UpdateV2("USER_DATA", gameDataRowInDate, Backend.UserInDate, param, callback =>
156             {
157                 if ( callback.IsSuccess() )
158                 {
159                     Debug.Log($"게임 정보 데이터 수정에 성공했습니다. : {callback}");
160
161                     action?.Invoke();
162                     onGameDataLoadEvent?.Invoke();
163                 }
164                 else...
168             });
169         }
170     }
171 }
```

우편에 포함된 아이템을 수령해서 BackendGameData.GameDataUpdate()
메소드가 호출되었을 때 Lobby 씬의 상단에 있는 재화 정보가 변경되도록 설정

우편 [하나/전체] 수령

- 결과 화면 (Console View)

Console

Clear | Collapse | Error Pause | Editor | 26 | 0 | 0

[19:10:18] 0번째 우편
title : 게임 오픈 기념 보상 2

[19:10:18] 1번째 우편
title : 게임 오픈 기념 보상 1

[19:10:20] Admin의 2023-06-19T10:10:08.009Z 우편수령을 요청합니다.
UnityEngine.Debug:Log (object)

[19:10:20] Admin의 2023-06-19T10:10:08.009Z 우편수령에 성공했습니다. : statusCode : 200
message : Success

[19:10:20] 재화차트 - GoodsChart.xlsx
UnityEngine.Debug:Log (object)

[19:10:20] [1] gold : 게임 내 아이템을 구매하는데 사용하는 재화, 획득 수량 : 500
UnityEngine.Debug:Log (object)

[19:10:20] 아이템을 수령했습니다. : gold - 500개
UnityEngine.Debug:Log (object)

[19:10:20] 2023-06-11T13:12:57.994Z의 게임 정보 데이터 수정을 요청합니다.
UnityEngine.Debug:Log (object)

[19:10:20] 게임 정보 데이터 수정에 성공했습니다. : statusCode : 204
message : Success

Admin의 2023-06-19T10:10:08.009Z 우편수령에 성공했습니다. : statusCode : 200
message : Success
returnValue : {"postItems":[{"item":{"itemId":"1","itemName":"gold","itemInfo":"게임 내 아이템을 구매하는데 사용하는 재화","chartFileName":"GoodsChart.xlsx","itemCount":500,"chartName":"재화차트"}}}

UnityEngine.Debug:Log (object)
BackendPostSystem/<>c__DisplayClass3_0:<PostReceive>b__0 (BackEnd.BackendReturnObject) (at Assets/Scripts/#100Backend/BackendPostSystem.cs:131)
#C.#B:Invoke ()
#C.#E:Poll ()
BackEnd.Backend:AsyncPoll ()

우편 [하나/전체] 수령

- 결과 화면 (Game View)

게임시작

게임시작

< 수령 전 >

< 수령 후 >

우편 [하나/전체] 수령

우편 전체 수령

- ReceivePostItemAll() 메소드를 이용한 모든 우편 수령
  - BackendPostSystem Script 수정

```
1   using System.Collections.Generic;
2   using UnityEngine;
3   using BackEnd;
4
5   public class BackendPostSystem : MonoBehaviour
6   {
7       private List<PostData> postList = new List<PostData>();
8
9       private void Update()
10      {
11          if ( Input.GetKeyDown("1") )...
15          else if ( Input.GetKeyDown("2") )...
19          else if ( Input.GetKeyDown("3") )
20          {
21              PostReceiveAll(PostType.Admin);
22          }
23      }
24
25      public void PostListGet(PostType postType)...
110
111     public void PostReceive(PostType postType, int index)...
153
```

우편 [하나/전체] 수령

- 모든 우편을 수령할 때 아이템의 JsonData

```
{
    "postItems": [
        [
            // 우편 하나에 첨부된 아이템이 하나일 때
            {
                “item”: { .. }, “itemCount”: 1000, “chartName”: “재화차트”
            },
        ]
        [
            // 우편 하나에 첨부된 아이템이 여러 개일 때
            {
                “item”: { .. }, “itemCount”: 1000, “chartName”: “재화차트”
            },
            {
                “item”: { .. }, “itemCount”: 500, “chartName”: “재화차트”
            },
        ]
    ]
}
```

우편 [하나/전체] 수령

- BackendPostSystem Script 수정 (계속)

```
154  public void PostReceiveAll(PostType postType)
155  {
156      if ( postList.Count <= 0 )
157      {
158          Debug.LogWarning("받을 수 있는 우편이 존재하지 않습니다. 혹은 우편 리스트 불러오기를 먼저 호출해주세요.");
159          return;
160      }
161
162      Debug.Log($"{postType.ToString()} 우편 전체 수령을 요청합니다.");
163
164      Backend.UPost.ReceivePostItemAll(postType, callback =>
165      {
166          if ( !callback.IsSuccess() )
167          {
168              Debug.LogError($"{postType.ToString()} 우편 전체 수령 중 에러가 발생했습니다. : {callback}");
169              return;
170          }
171
172          Debug.Log($"우편 전체 수령에 성공했습니다. : {callback}");
173
174          postList.Clear();       // 모든 우편을 수령했기 때문에 postList는 초기화한다
175
176          // 모든 우편의 아이템 저장
177          foreach ( LitJson.JsonData postItemsJson in callback.GetFlattenJSON()["postItems"] )
178          {
179              SavePostToLocal(postItemsJson);
180          }
181
182          // 플레이어의 재화 정보를 서버에 업데이트
183          BackendGameData.Instance.GameDataUpdate();
184      });
185  }
```

Tip. 0번 우편의 아이템 = callback.GetFlattenJSON()["postItems"][0]

우편 [하나/전체] 수령

- 결과 화면 (Console View)

Console

Clear | Collapse | Error Pause | Editor | 27 | 0 | 0

[19:26:59] 0번째 우편
title : 게임 오픈 기념 보상 2

[19:26:59] 1번째 우편
title : 게임 오픈 기념 보상 1

[19:27:03] Admin 우편 전체 수령을 요청합니다.
UnityEngine.Debug:Log (object)

[19:27:03] 우편 전체 수령에 성공했습니다. : statusCode : 200
message : Success

[19:27:03] 재화차트 - GoodsChart.xlsx
UnityEngine.Debug:Log (object)

[19:27:03] [1] gold : 게임 내 아이템을 구매하는데 사용하는 재화, 획득 수량 : 500
UnityEngine.Debug:Log (object)

[19:27:03] 아이템을 수령했습니다. : gold - 500개
UnityEngine.Debug:Log (object)

[19:27:03] 재화차트 - GoodsChart.xlsx
UnityEngine.Debug:Log (object)

[19:27:03] [1] gold : 게임 내 아이템을 구매하는데 사용하는 재화, 획득 수량 : 1000
UnityEngine.Debug:Log (object)

[19:27:03] 아이템을 수령했습니다. : gold - 1000개
UnityEngine.Debug:Log (object)

[19:27:03] 2023-06-11T13:13:16.073Z의 게임 정보 데이터 수정을 요청합니다.
UnityEngine.Debug:Log (object)

[19:27:03] 게임 정보 데이터 수정에 성공했습니다. : statusCode : 204
message : Success

0번째 우편의 gold 500과 1번째 우편의 gold 1000을 모두 받아 총 gold 1500 획득

우편 전체 수령에 성공했습니다. : statusCode : 200
message : Success
returnValue : {"postItems":[[{"item":{"itemId":"1","itemName":"gold","itemInfo":"게임 내 아이템을 구매하는데 사용하는 재화","chartFileName":"GoodsChart.xlsx","itemCount":500,"chartName":"재화차트"},[{"item":{"itemId":"1","itemName":"gold","itemInfo":"게임 내 아이템을 구매하는데 사용하는 재화","chartFileName":"GoodsChart.xlsx","itemCount":1000,"chartName":"재화차트"}]]}

UnityEngine.Debug:Log (object)
BackendPostSystem/<>c__DisplayClass4_0:<PostReceiveAll>b__0 (BackEnd.BackendReturnObject) (at Assets/Scripts/#100Backend/BackendPostSystem.cs:172)
#C.#B:Invoke ()
#C.#E:Poll ()
BackEnd.Backend:AsyncPoll ()
BackendManager:Update () (at Assets/Scripts/#100Backend/BackendManager.cs:21)

우편 [하나/전체] 수령

- 결과 화면 (Console View)

1923 + 1500 = 3423

게임시작

게임시작

< 수령 전 >

< 수령 후 >