



section3-class01-Google Tasks API

구현을 위한 사전 지식

✓ Google Tasks API 소개

1 Google Tasks API란?

홈 > Google Workspace > Google Tasks > 참고자료 도움이 되었나요? 👍 🗨

Google Tasks API 📄 ▾ 의견 보내기

Google Tasks API를 사용하면 할 일 및 할 일 목록을 관리할 수 있습니다.

서비스: tasks.googleapis.com

이 서비스를 호출하려면 Google에서 제공하는 클라이언트 라이브러리를 사용하는 것이 좋습니다. 애플리케이션이 이 서비스를 호출하기 위해 자체 라이브러리를 사용해야 하는 경우, API 요청을 할 때 다음 정보를 사용합니다.

Google Tasks API 참고: <https://developers.google.com/tasks/reference/rest?hl=ko>

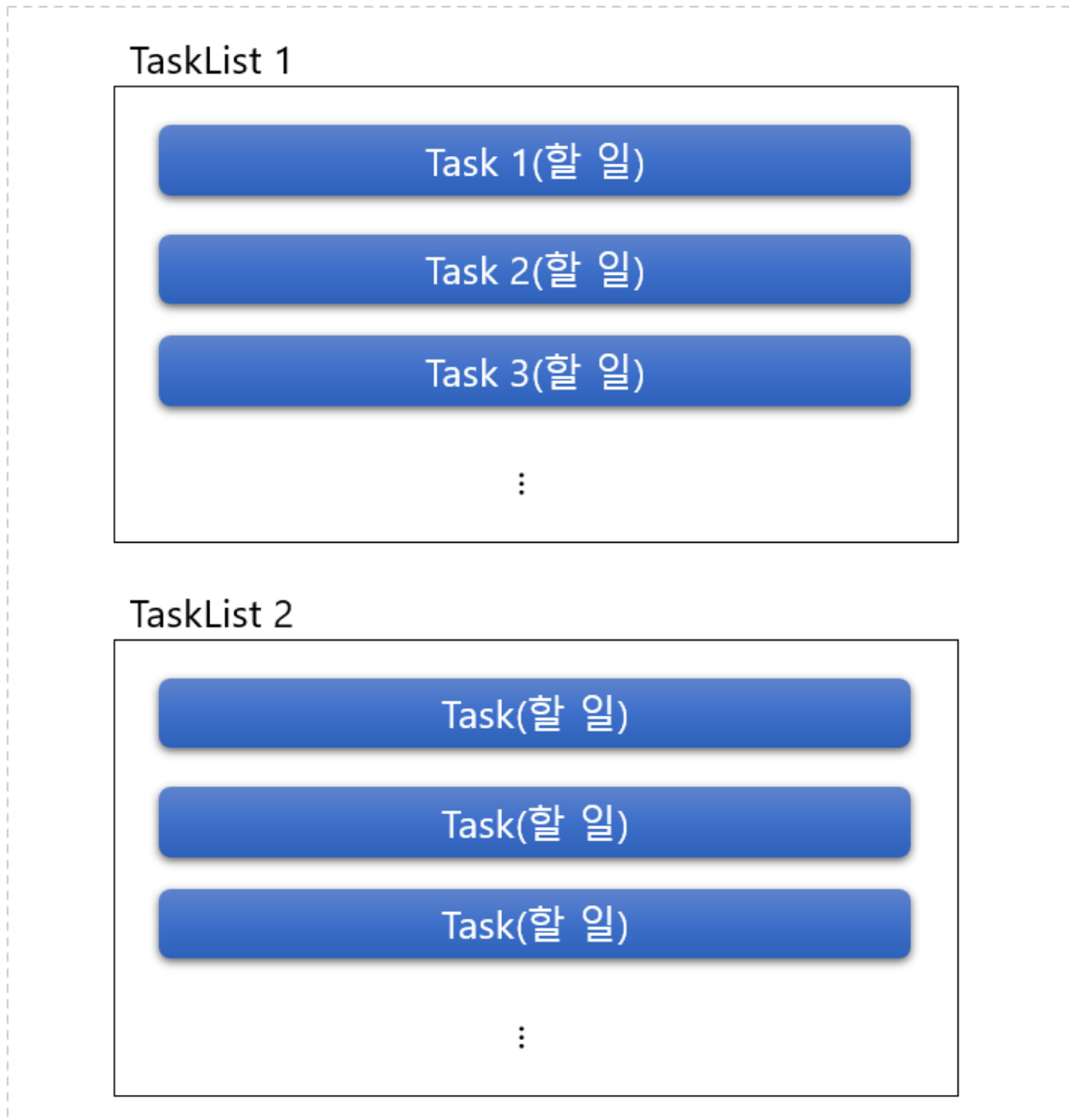
2 Google Tasks API에서 알아야 하는 구성요소

✓ TaskList

: 할 일이 포함된 할일 목록을 표현하는 클래스. 즉, TaskList는 한 개의 할 일 목록을 의미한다.

✓ Task

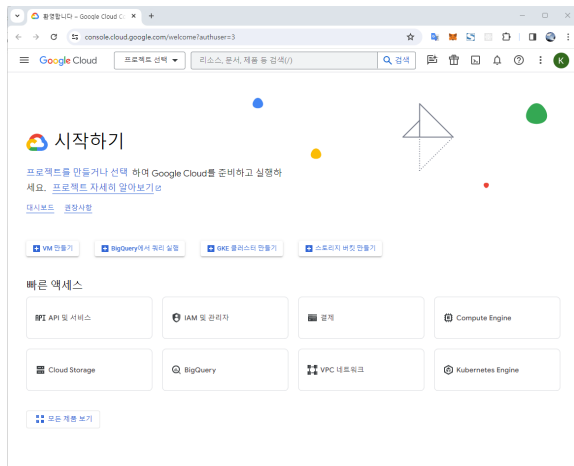
: 할 일 목록에 포함된 할 일을 표현하는 클래스. 즉, Task는 한 개의 할 일을 의미한다.



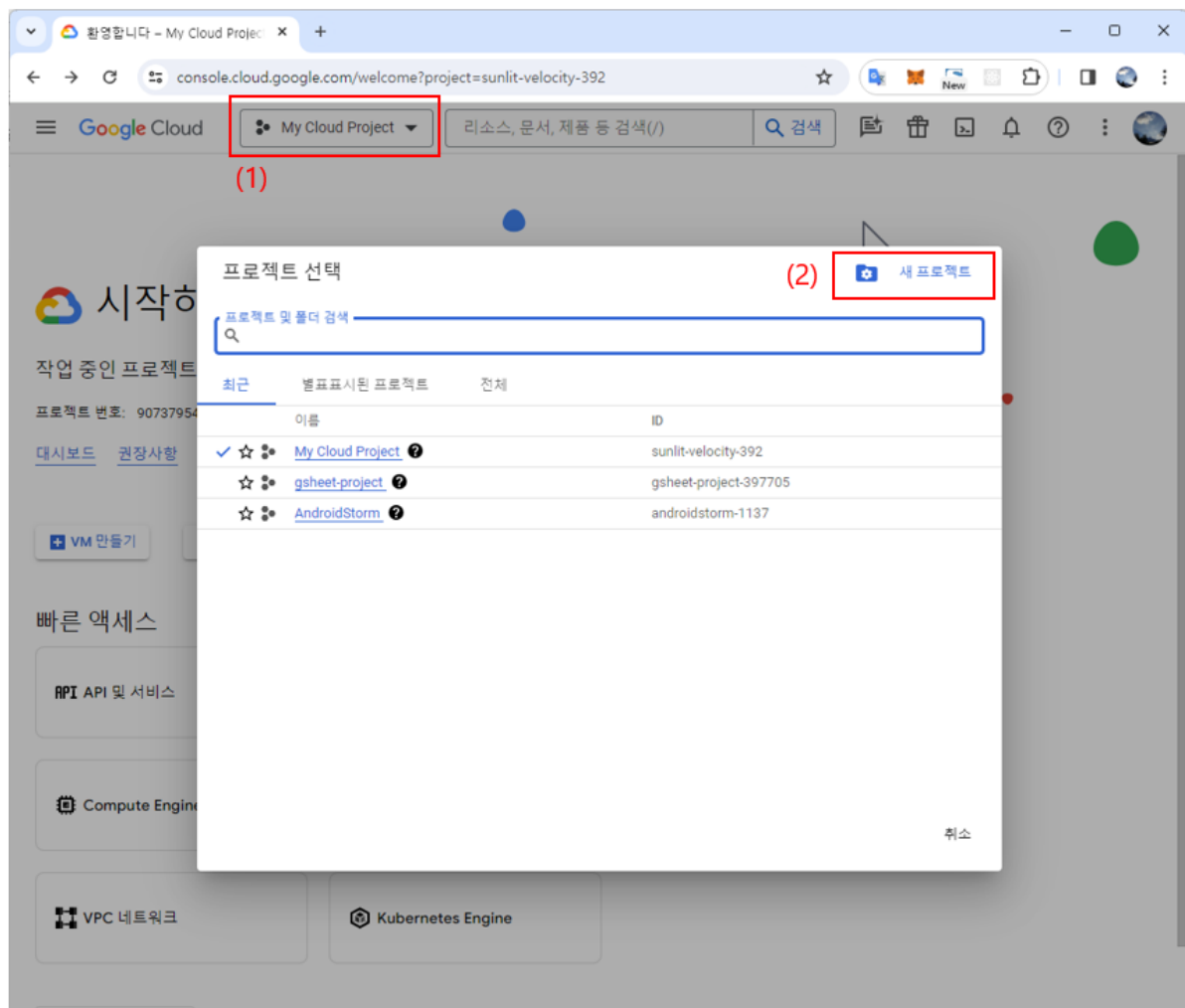
TaskList와 Task의 관계를 표현한 그림

3 Google Tasks API 사용을 위한 사전 준비 작업

<https://console.cloud.google.com/> 에서 Google의 API 서비스를 이용하기 위한 사전 설정 작업이 필요하다.

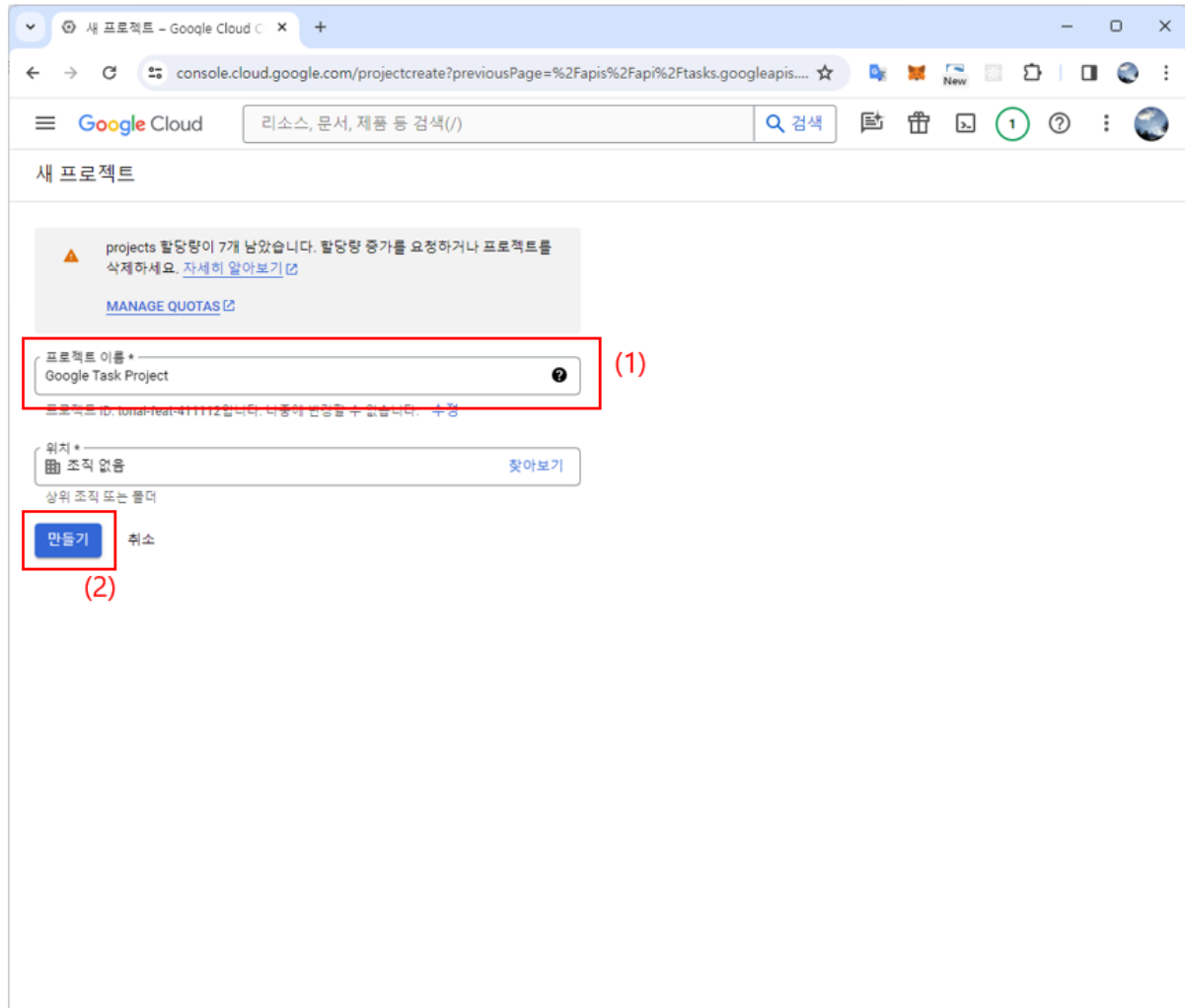


✓ Google Tasks API를 사용할 프로젝트 생성



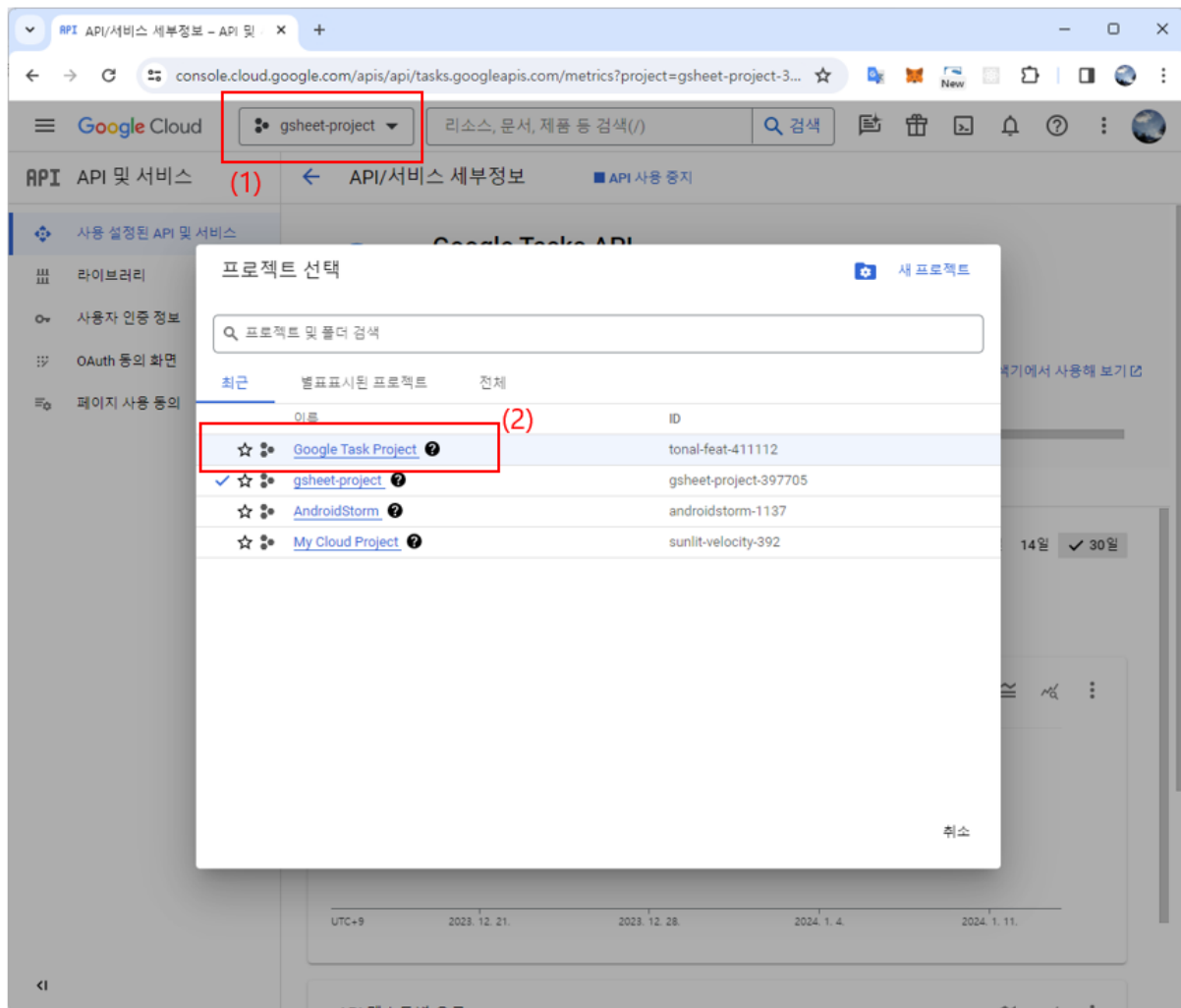
[그림 3-1] 프로젝트 생성

[그림 3-1]의 (1)의 프로젝트 선택 박스를 클릭한 후, (2)와 같이 새 프로젝트를 클릭합니다.



[그림 3-2] 프로젝트 만들기

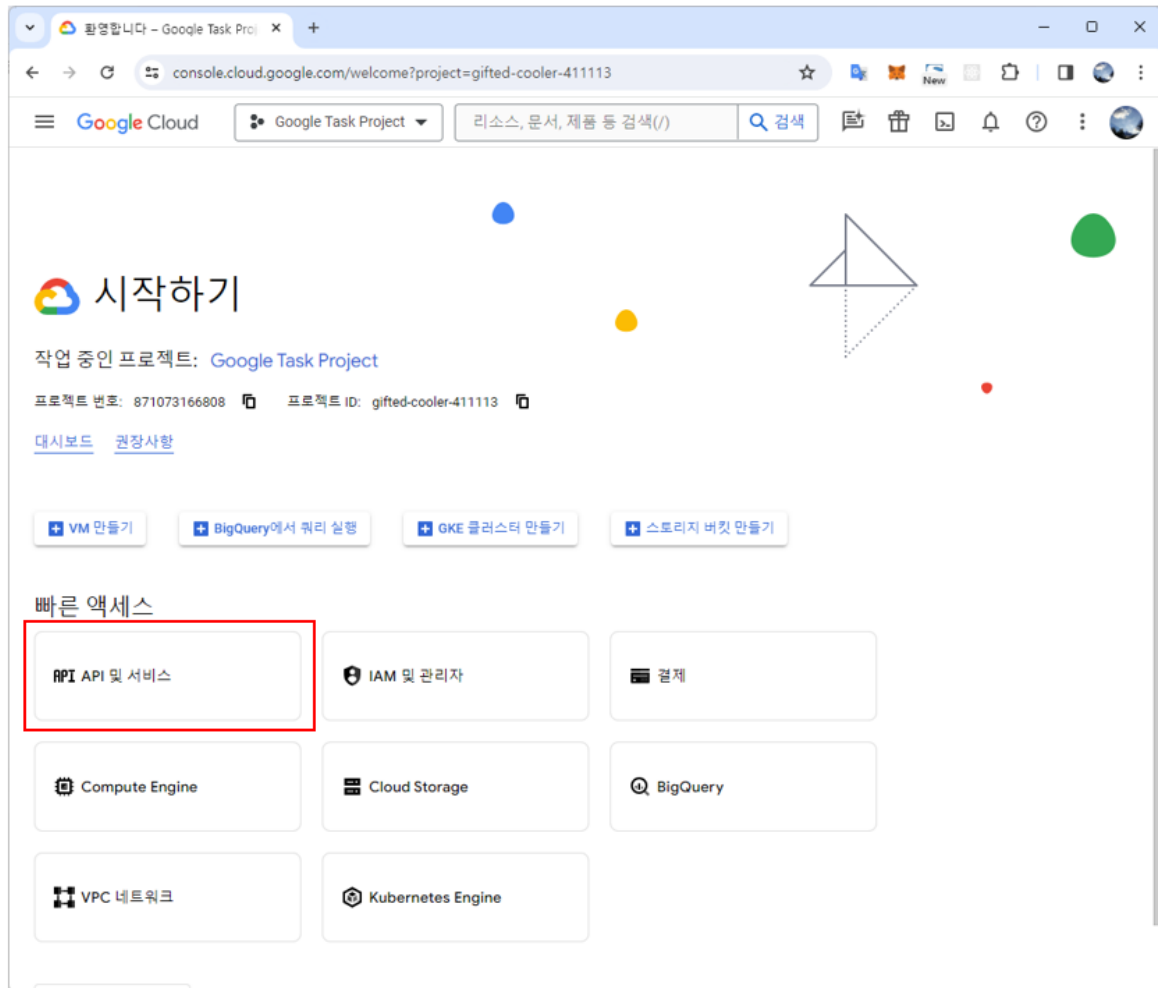
[그림 3-2]의 (1)과 같이 프로젝트 이름을 적절하게 입력한 후, (2)의 **만들기** 버튼을 클릭합니다.



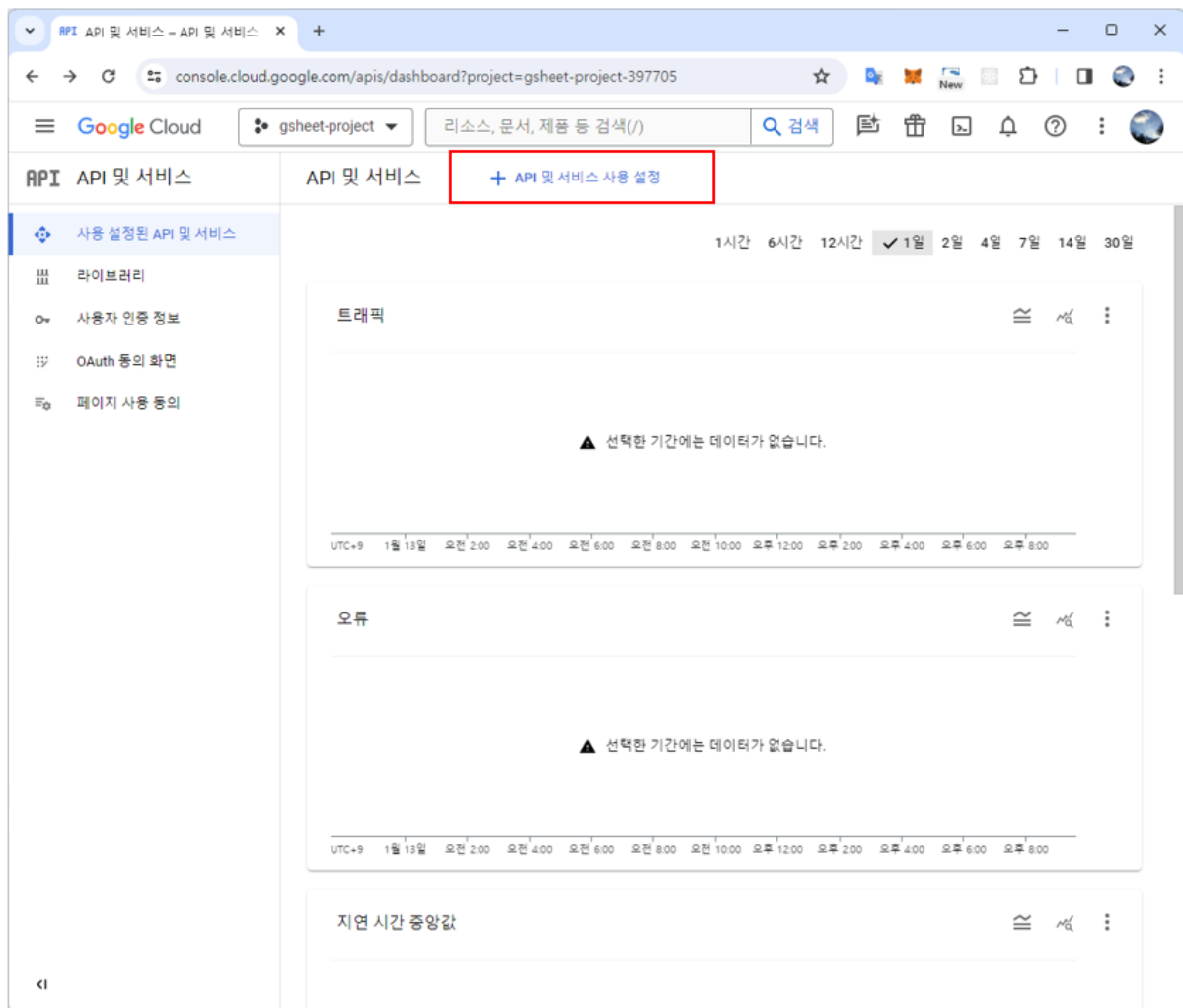
[그림 3-3] Google Task Project 선택

[그림 3-3]과 같이 앞에서 생성한 프로젝트를 선택합니다.

✓ Google Tasks API 서비스 활성화

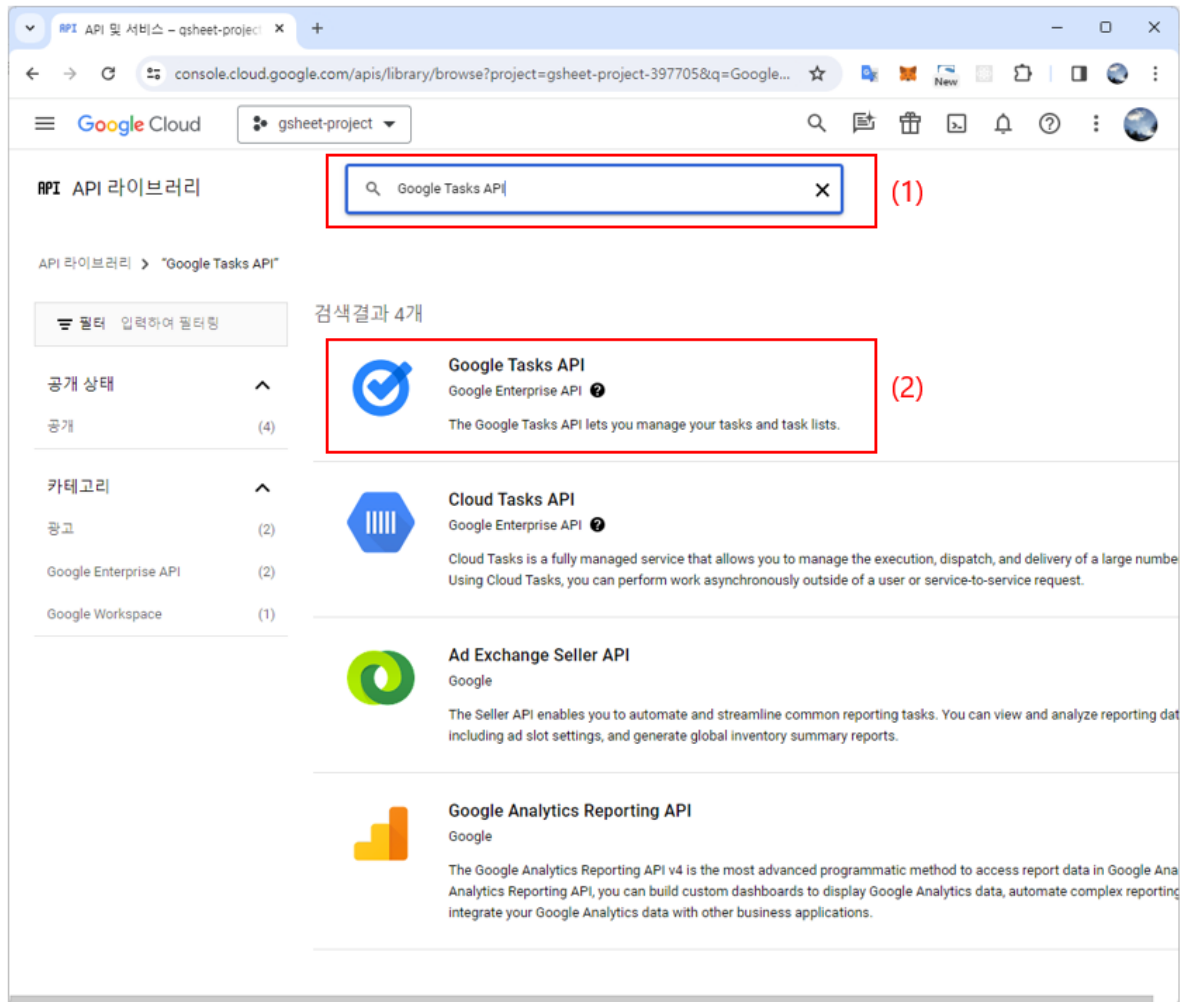


[그림 3-4]와 같이 **API 및 서비스** 메뉴를 클릭합니다.



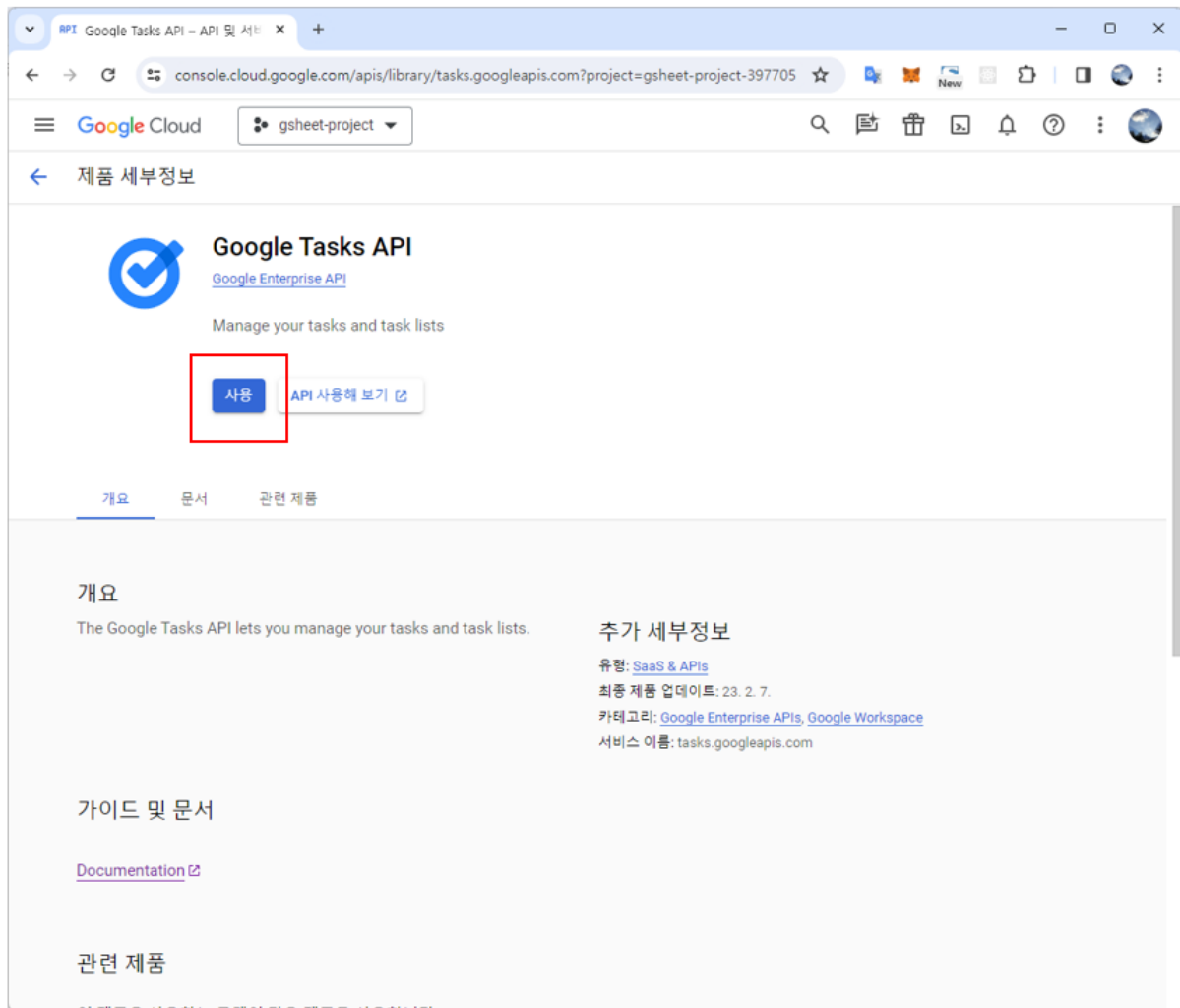
[그림 3-5] API 및 서비스 사용 설정 추가

[그림 3-5]와 같이 **API 및 서비스 사용 설정** 을 클릭합니다.



[그림 3-6] Google Tasks API 검색

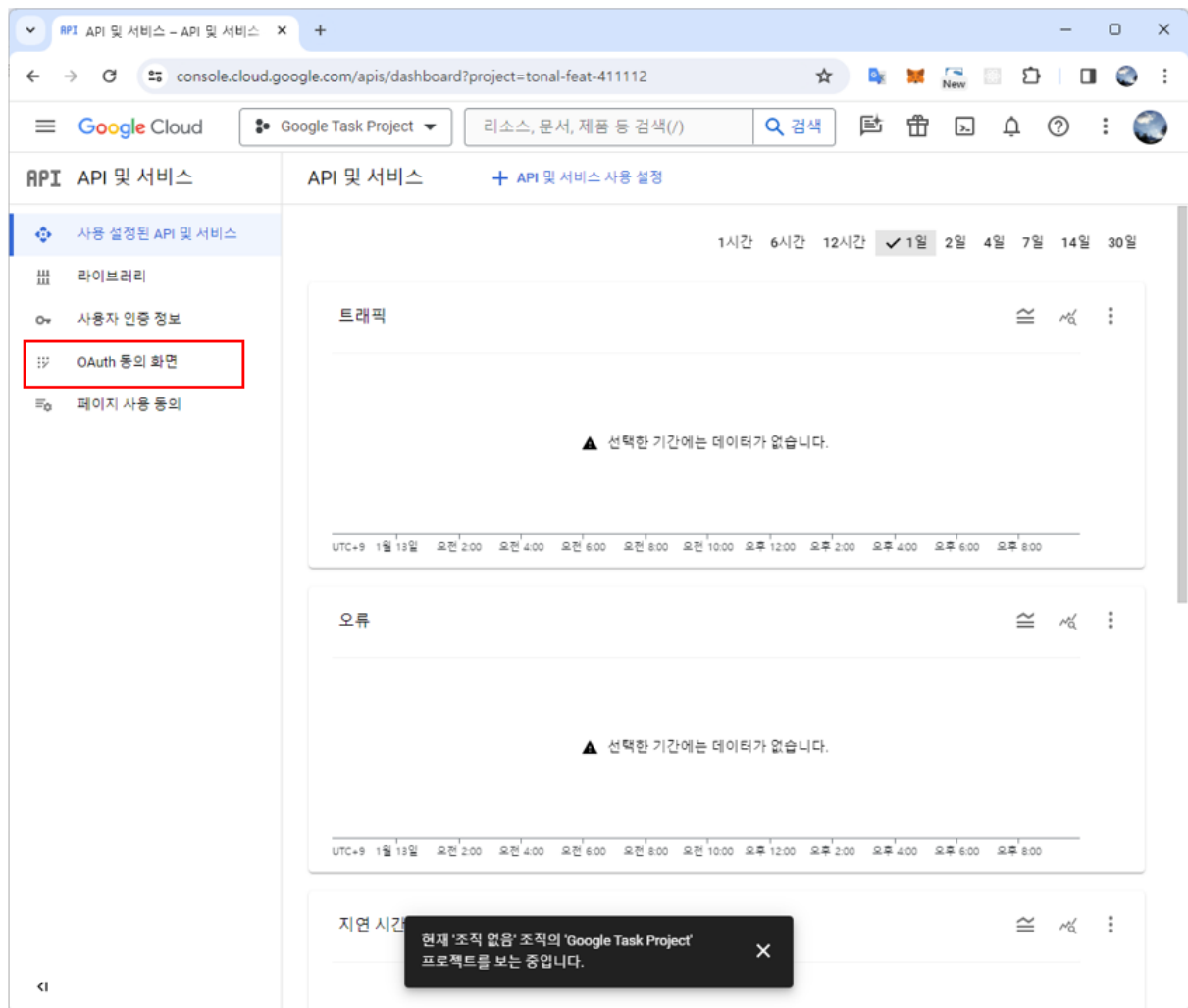
[그림 3-6]의 (1)과 같이 **Google Tasks API** 를 검색한 후, (2)와 같이 **Google Tasks API** 를 클릭합니다.



[그림 3-7] Google Tasks API 활성화

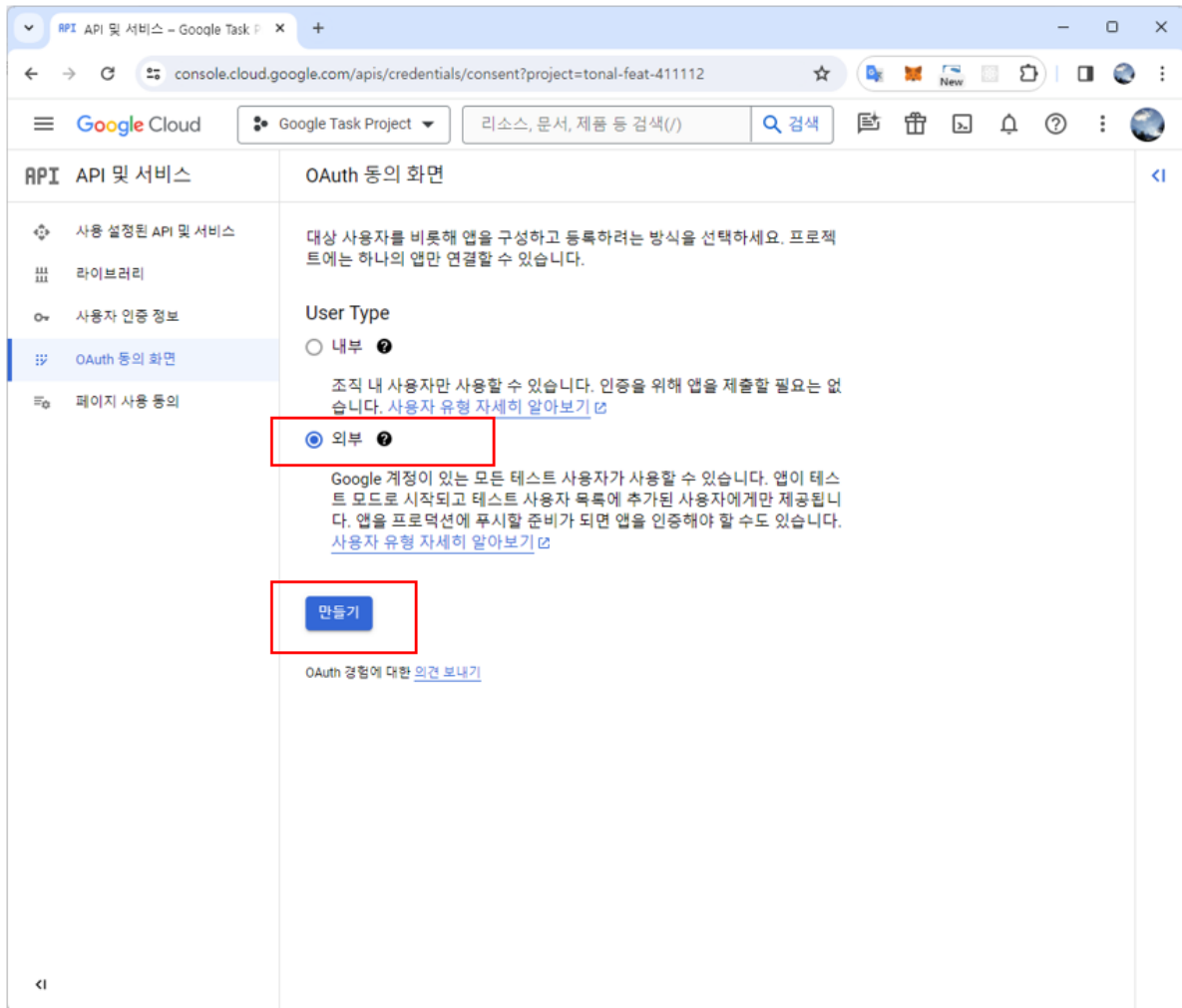
[그림 3-7]와 같이 **사용** 버튼을 클릭해 Google Tasks API를 활성화 시킵니다.

✓ OAuth 동의 화면 구성



[그림 3-8] OAuth 동의 화면 메뉴 선택

[그림 3-8]과 같이 **OAuth 동의 화면** 메뉴를 클릭합니다.



[그림 3-9] User Type 선택

[그림 3-9]와 같이 User Type을 **외부** 로 체크 한 후, **만들기** 버튼을 클릭합니다.

API 앱 등록 수정 - API 및 서비스 - X

console.cloud.google.com/apis/credentials/consent/edit/newAppInternalUser=false?project...

Google Cloud Google Task Project 리소스, 문서, 제품 등 검색(/) 검색

API API 및 서비스

사용 설정된 API 및 서비스 라이브러리 사용자 인증 정보 OAuth 동의 화면 페이지 사용 동의

앱 등록 수정

1 OAuth 동의 화면 — 2 범위 — 3 테스트 사용자 — 4 요약

앱 정보

동의 화면에 표시되어 최종 사용자가 개발자를 확인하고 문의할 수 있습니다.

앱 이름 *
Google Tasks API 애플리케이션
동의를 요청하는 앱의 이름

사용자 지원 이메일 *
-----@gmail.com
사용자가 동의에 대해 문의하는 용도입니다. [자세히 알아보기](#)

앱 로고

이 로고는 사용자가 앱을 알아보는 데 도움이 되고 OAuth 동의 화면에 표시될 수 있습니다. 앱이 내부 전용으로 구성되었거나 게시 상태가 '테스트 중'이 아니라면 로고를 업로드한 후 인증을 위해 앱을 제출해야 합니다. [자세히 알아보기](#)

업로드할 로고 파일 [찾아보기](#)

사용자가 앱을 알아보는 데 도움이 되도록 동의 화면에 대한 이미지(1MB 이하 크기)를 업로드합니다. 허용되는 이미지 형식은 JPG, PNG, BMP입니다. 최적의 결과를 위해서는 로고가 120x120px 크기의 정사각형이어야 합니다.

앱 도메인

나와 내 사용자를 보호하기 위해 Google에서는 OAuth를 사용하는 앱만 승인된 도메인을 이용할 수 있도록 허용합니다. 다음 정보가 동의 화면에서 사용자에게 표시됩니다.

애플리케이션 홈페이지
사용자에게 홈페이지 링크를 제공합니다.

[그림 3-10] 앱 정보 입력

[그림 3-10]과 같이 **앱 이름** 과 **사용자 지원 이메일** 항목을 채워 넣습니다.

API 앱 등록 수정 - API 및 서비스 - x +

console.cloud.google.com/apis/credentials/consent/edit/newAppInternalUser=false?project...

Google Cloud Google Task Project 리소스, 문서, 제품 등 검색(/) 검색

API API 및 서비스

- 사용 설정된 API 및 서비스
- 라이브러리
- 사용자 인증 정보
- OAuth 동의 화면**
- 페이지 사용 동의

앱 등록 수정

도움말: 허용되는 이미지 형식은 JPG, PNG, BMP입니다. 최적의 결과를 위해서는 로고가 120x120px 크기의 정사각형이어야 합니다.

앱 도메인

나와 내 사용자를 보호하기 위해 Google에서는 OAuth를 사용하는 앱만 승인된 도메인을 이용할 수 있도록 허용합니다. 다음 정보가 동의 화면에서 사용자에게 표시됩니다.

애플리케이션 홈페이지
사용자에게 홈페이지 링크를 제공합니다.

애플리케이션 개인정보처리방침 링크
사용자에게 공개 개인정보처리방침 링크를 제공합니다.

애플리케이션 서비스 약관 링크
사용자에게 공개 서비스 약관 링크를 제공합니다.

승인된 도메인 ⓘ

동의 화면 또는 OAuth 클라이언트 구성에서 도메인이 사용되면 여기에서 사전 등록해야 합니다. 앱이 인증을 거쳐야 하는 경우 [Google Search Console](#)로 이동하여 도메인이 승인되었는지 확인하세요. 승인된 도메인 한도에 대해 [자세히 알아보세요](#).

+ 도메인 추가

개발자 연락처 정보

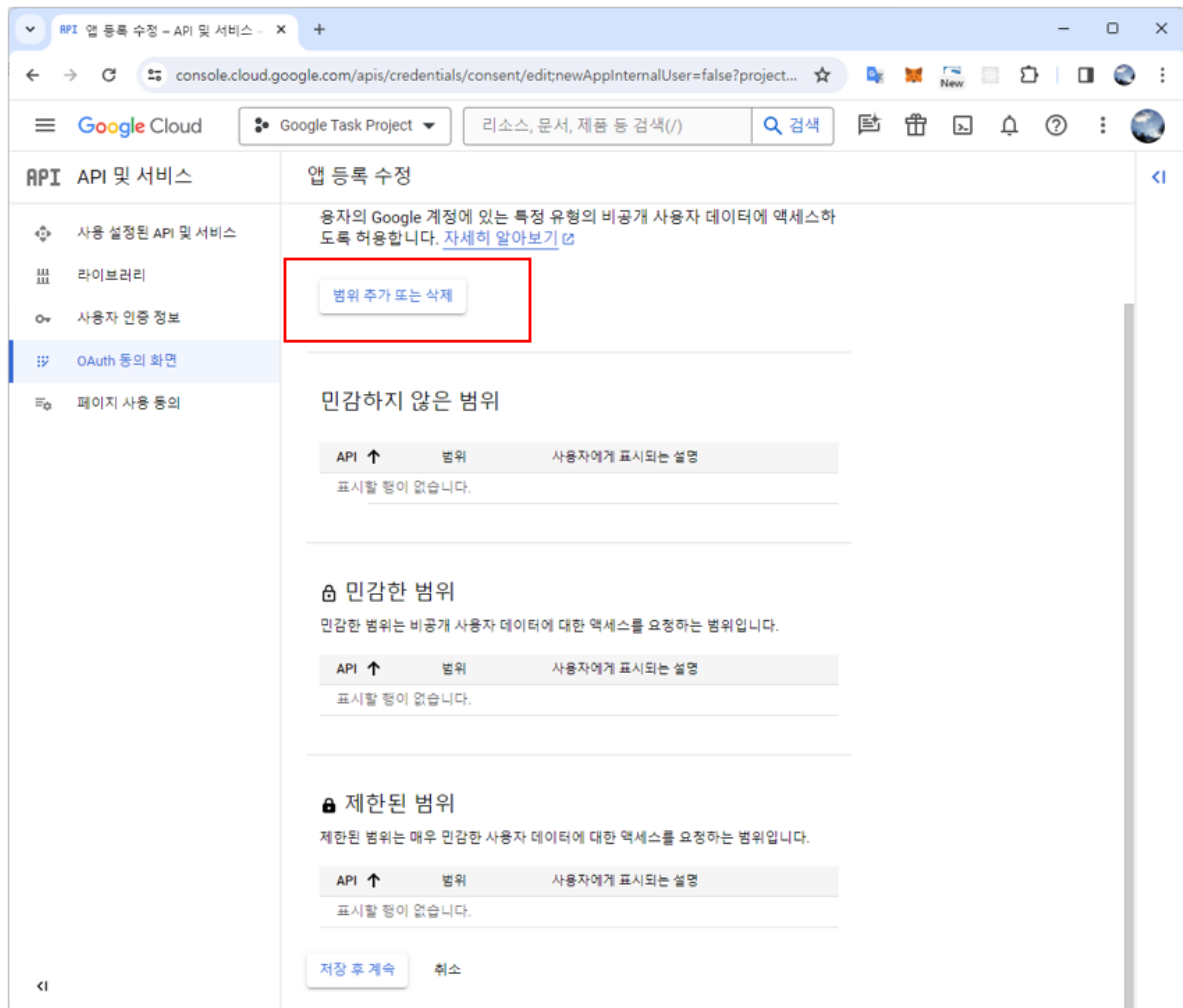
이메일 주소 *

이 이메일 주소는 Google에서 프로젝트 변경사항에 대해 알림을 보내기 위한 용도입니다.

저장 후 계속 취소

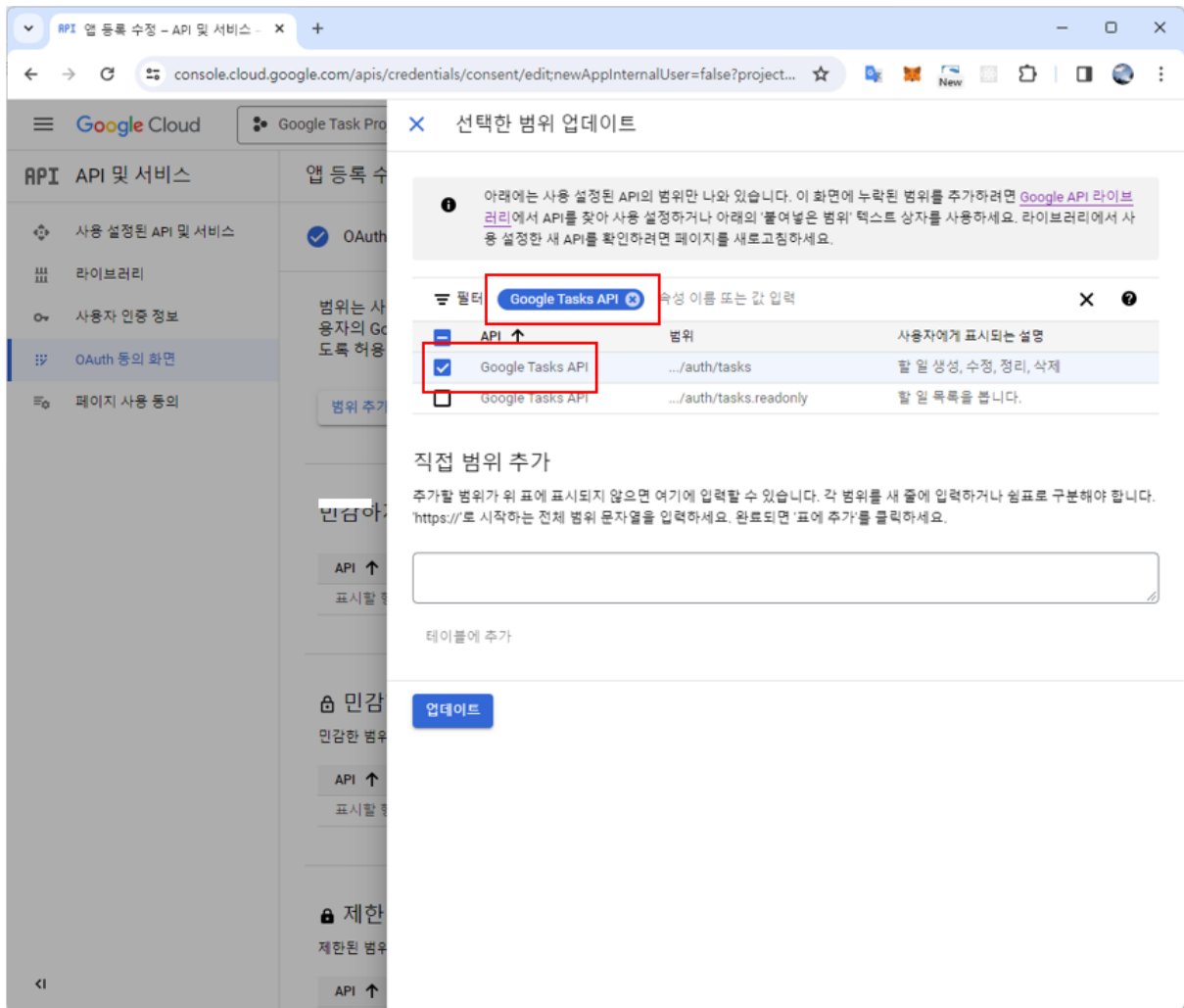
[그림 3-11] 앱 정보 입력 나머지

[그림 3-11]과 같이 앱 정보 입력 화면에서 **개발자 연락처 정보**의 이메일 주소를 입력한 뒤 **저장 후 계속** 버튼을 클릭합니다.



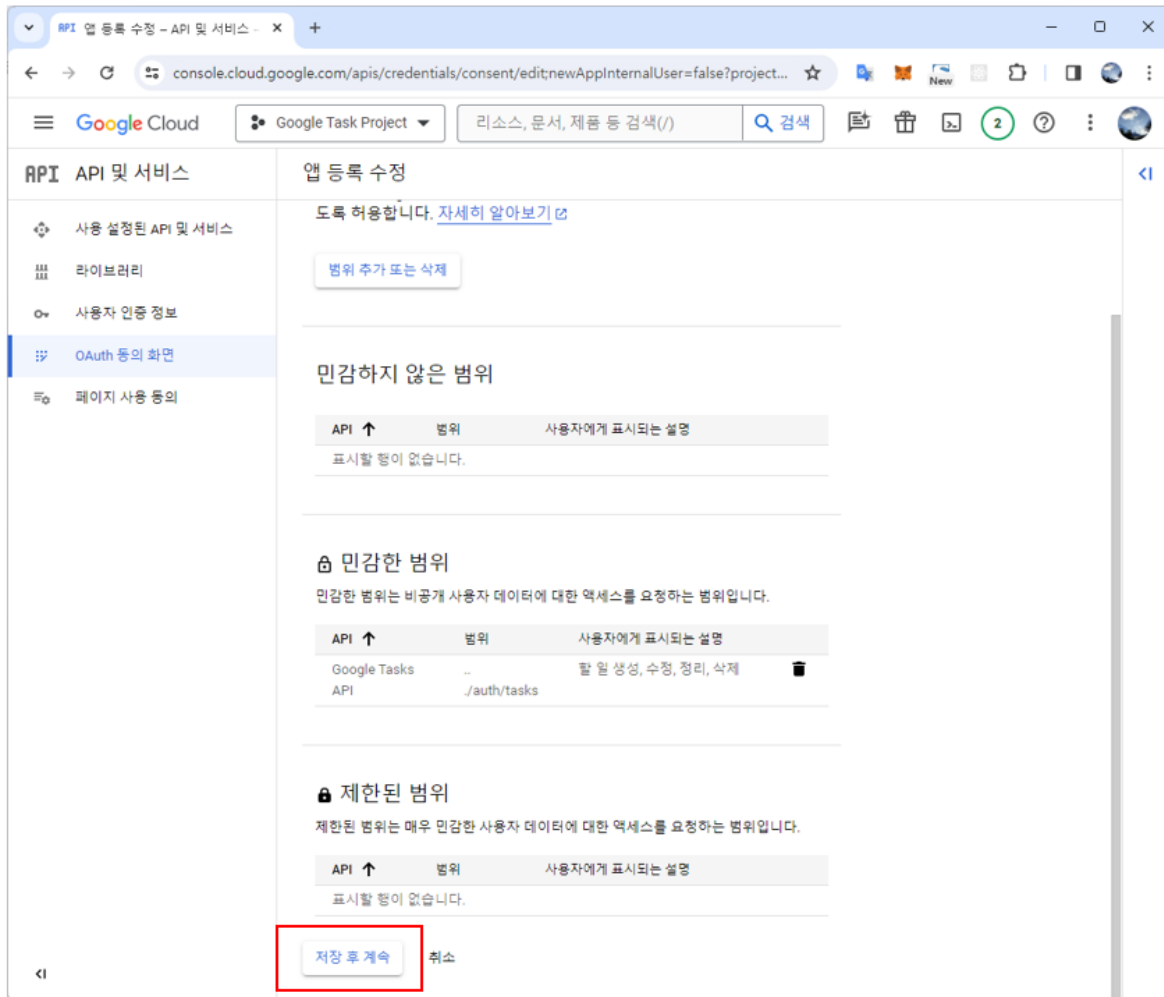
[그림 3-12] 범위 추가

[그림 3-12]와 같이 **범위 추가 또는 삭제** 버튼을 클릭합니다.



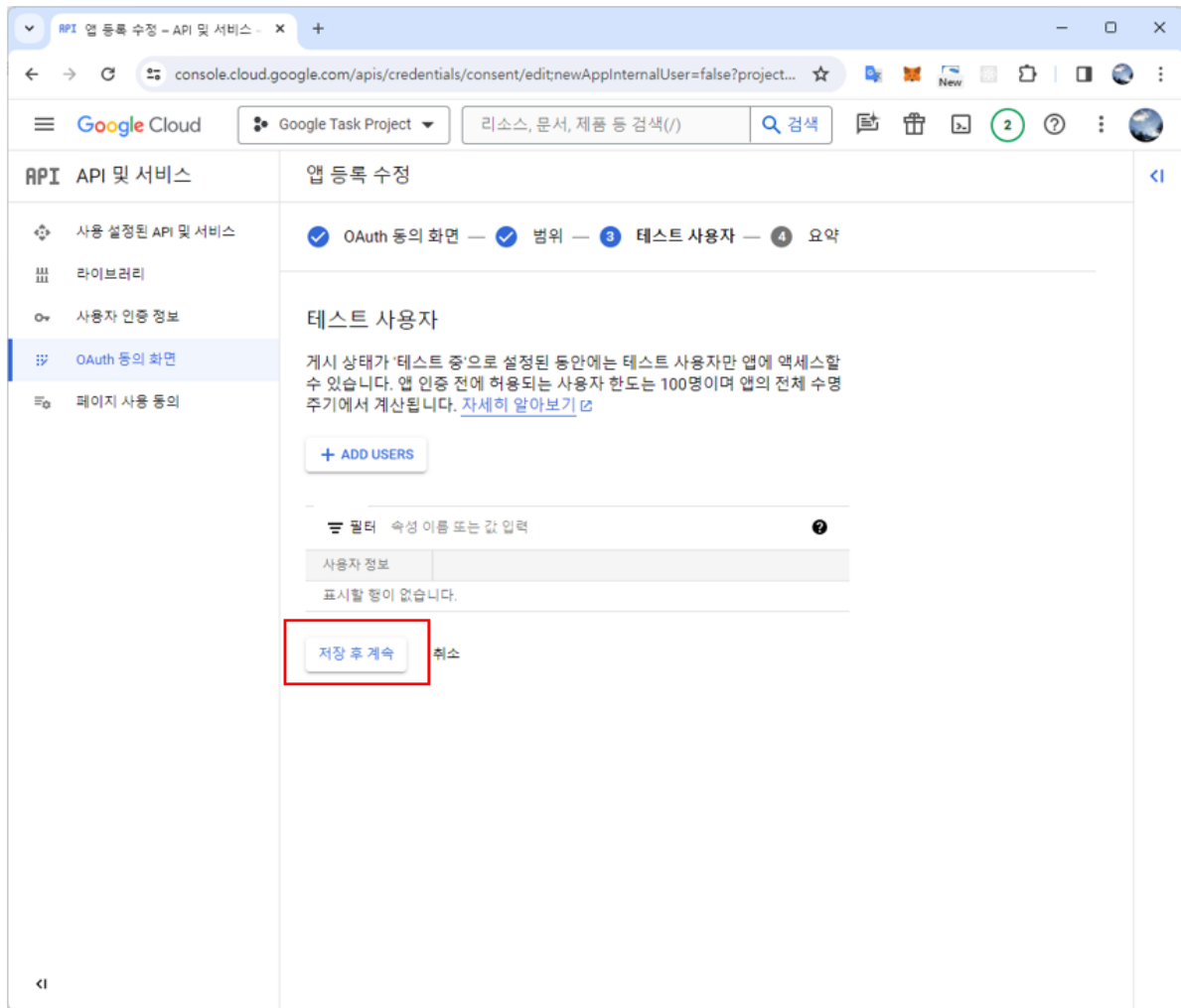
[그림 3-13] 사용 범위 업데이트

[그림 3-13]과 같이 필터 입력란에 **Google Tasks API** 를 입력한 후, 엔터를 칩니다. 그리고 검색 결과 목록에서 첫 번째 항목(범위: ../auth/tasks)을 체크 한 후, **업데이트** 버튼을 클릭 합니다.



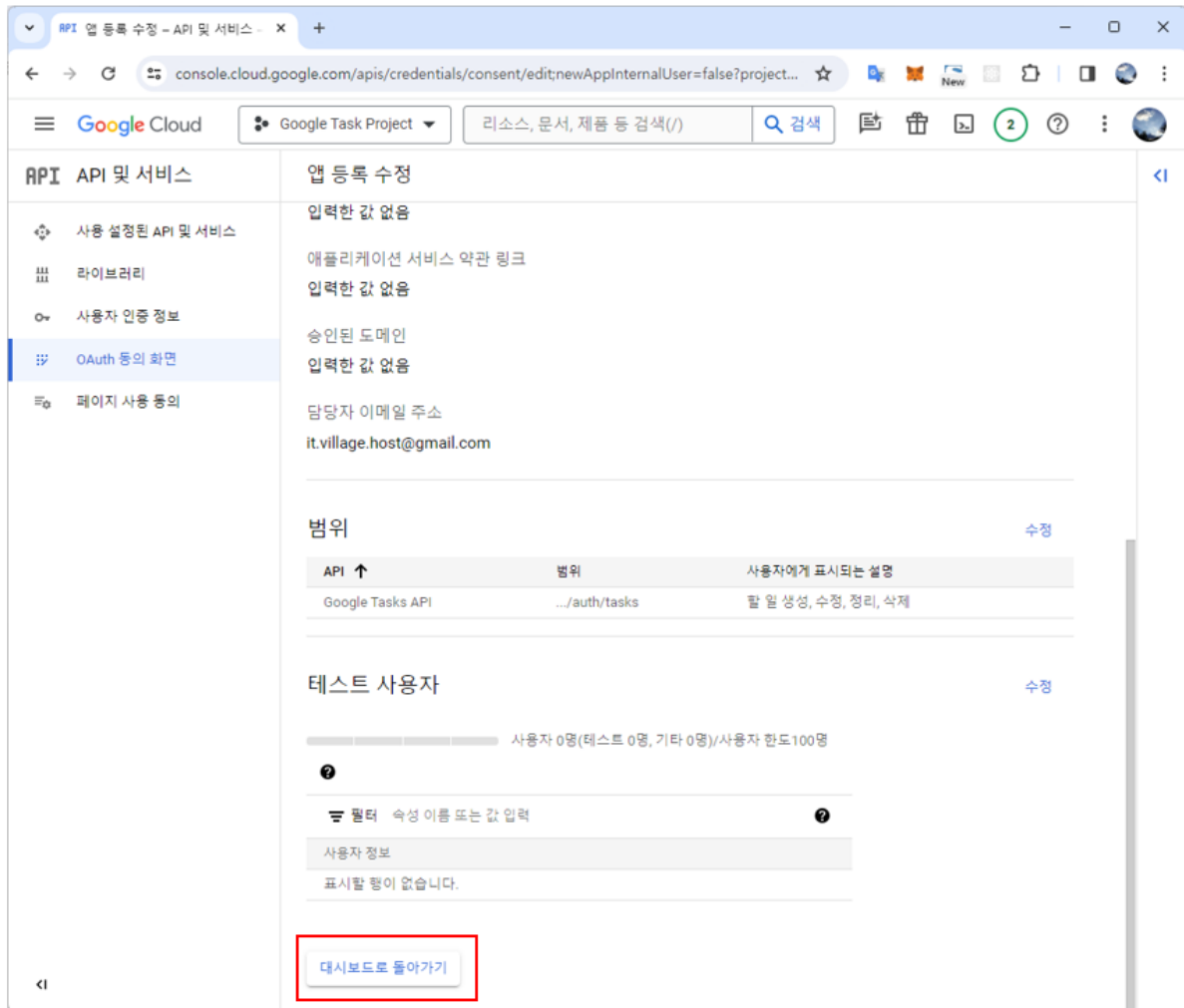
[그림 3-14] 저장 후 계속

[그림 3-14]와 같이 저장 후 계속 버튼을 클릭합니다.



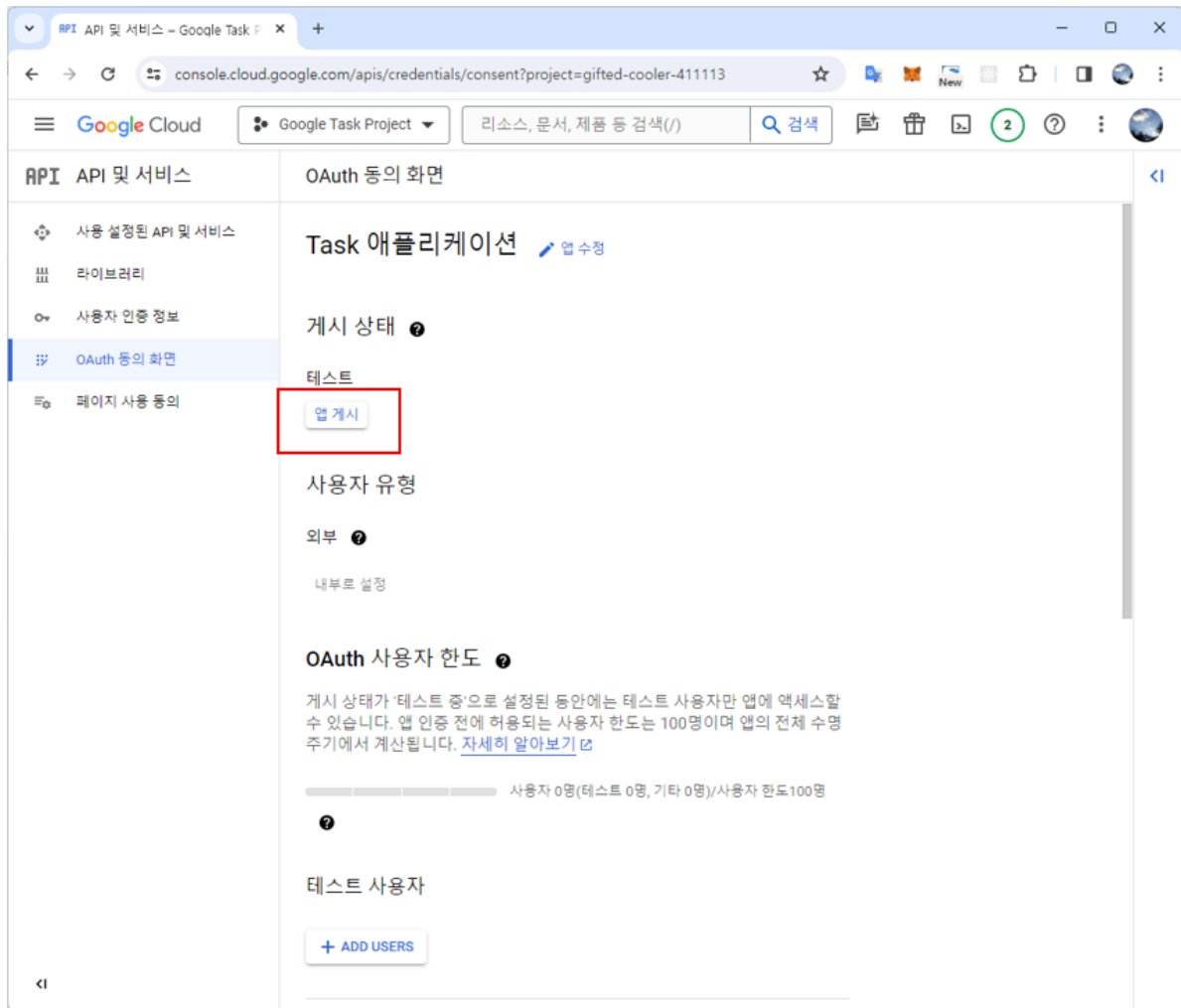
[그림 3-15] 테스트 사용자 화면

[그림 3-15]의 테스트 사용자 화면에서 **저장 후 계속** 버튼을 클릭합니다.



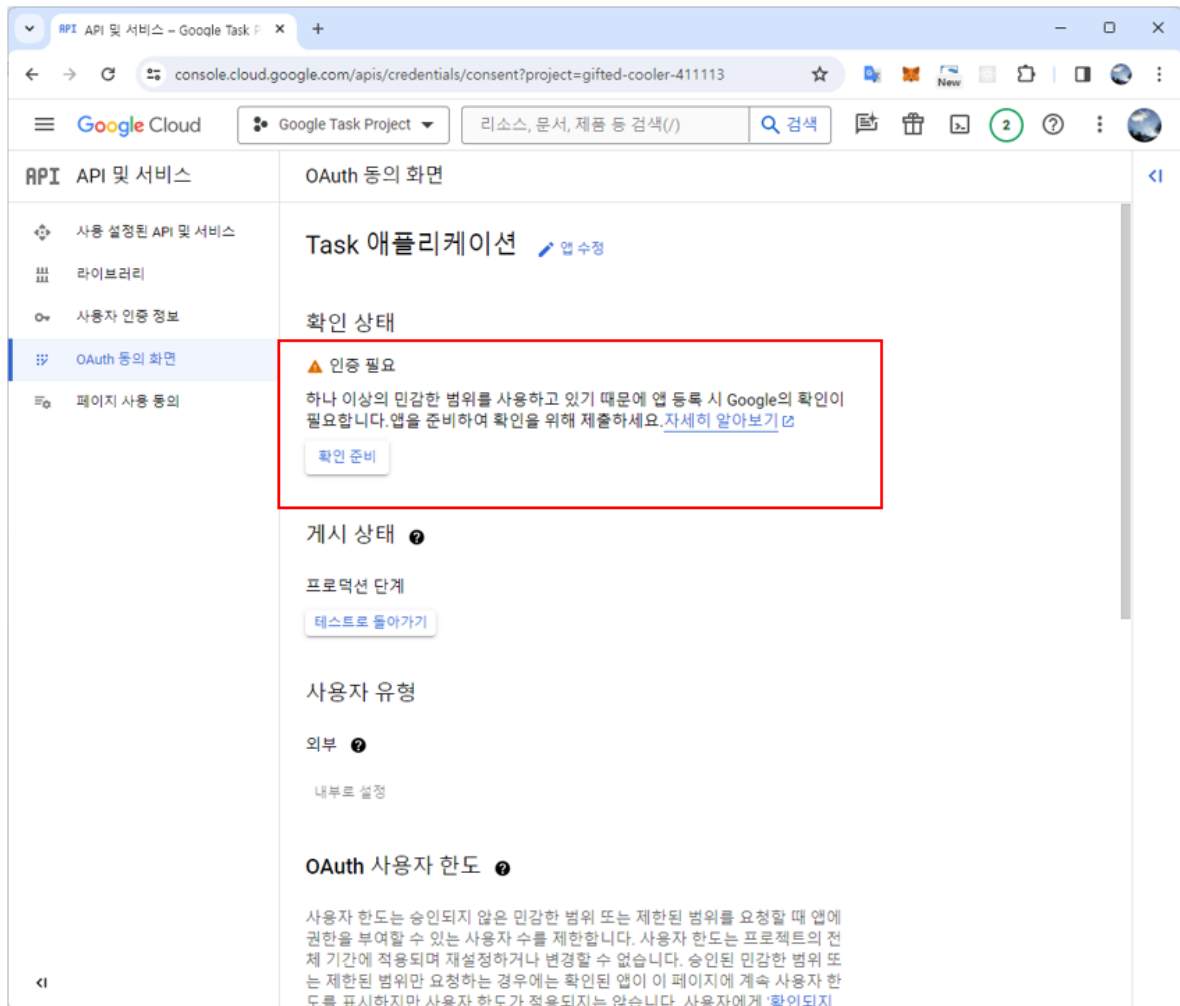
[그림 3-16] 요약 화면

[그림 3-16]과 같이 **요약** 화면에서 맨 아래쪽으로 스크롤을 내린 후, **대시보드로 돌아가기** 버튼을 클릭합니다.



[그림 3-17] OAuth 동의 화면 대시보드

[그림 3-17]과 같이 OAuth 동의 화면 대시보드에서 **앱 게시** 버튼을 클릭합니다.

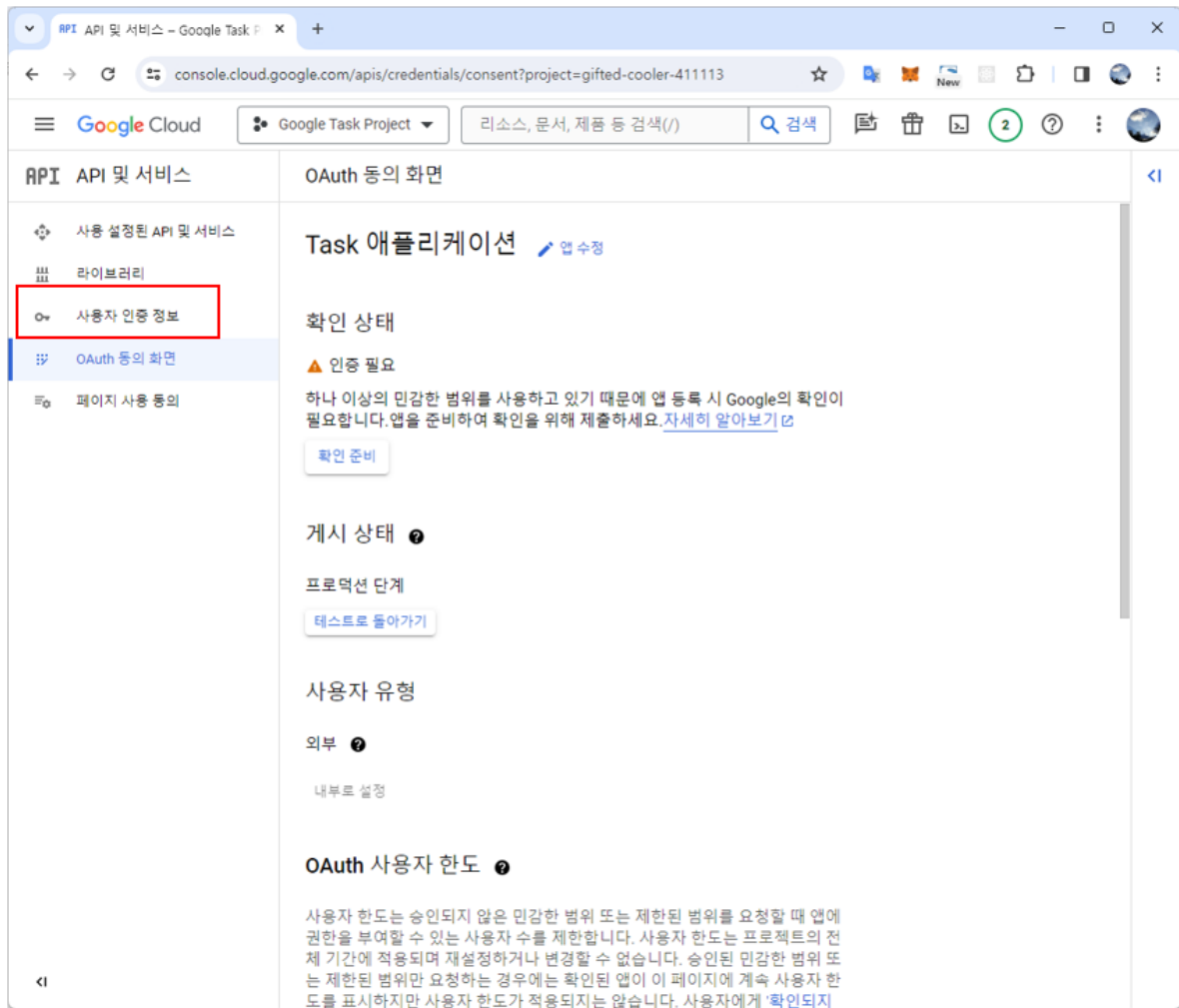


[그림 3-18] 확인 상태

앱을 게시하면 [그림 3-18]과 같이 **인증 필요** 와 같은 경고 메시지가 출력이 되는데 우리는 학습 목적으로 사용하기 때문에 별도의 인증을 할 필요가 없다는 사실을 기억하시기 바랍니다.

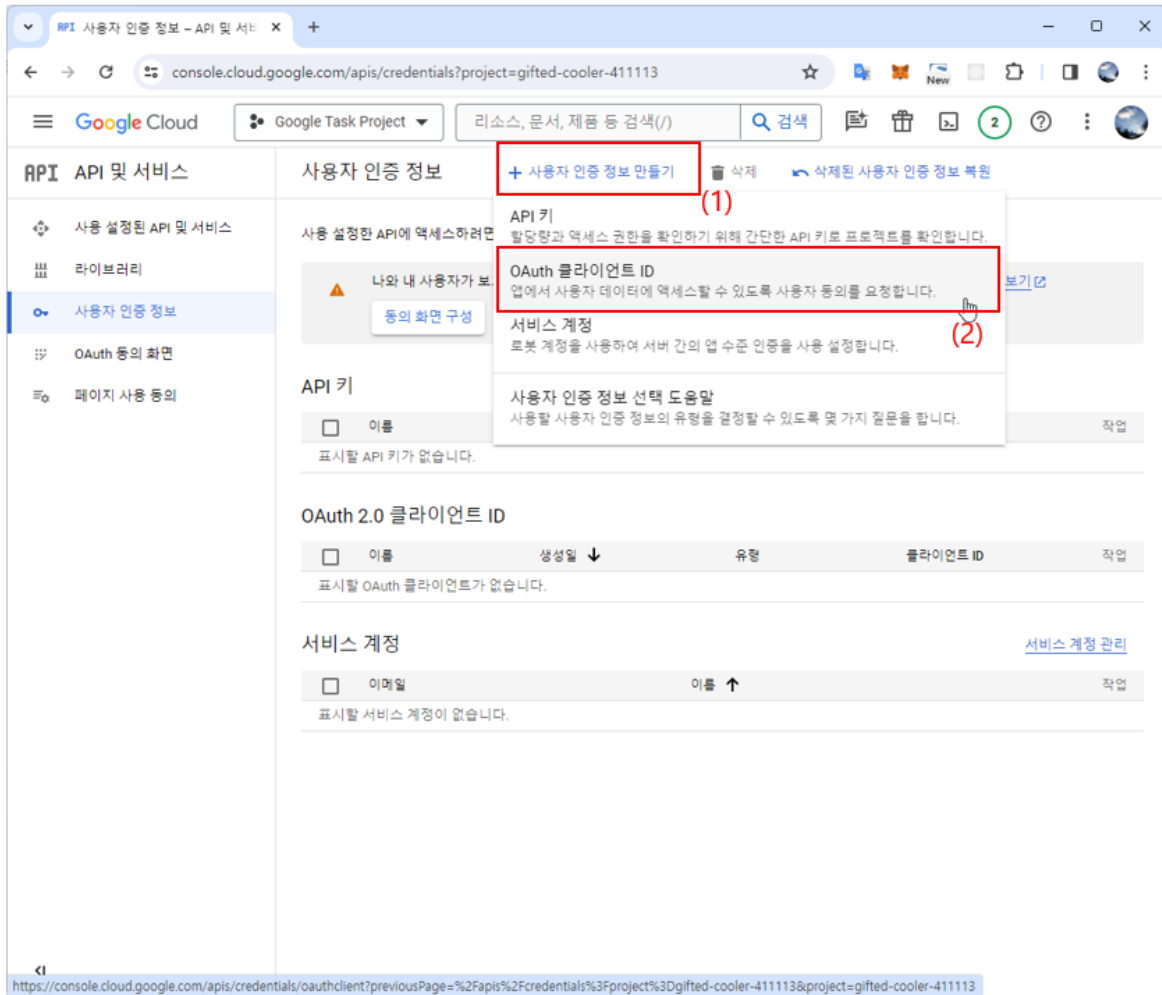
✓ 사용자 인증 정보 생성

이제 애플리케이션에서 사용할 사용자 인증 정보를 생성할 차례입니다.



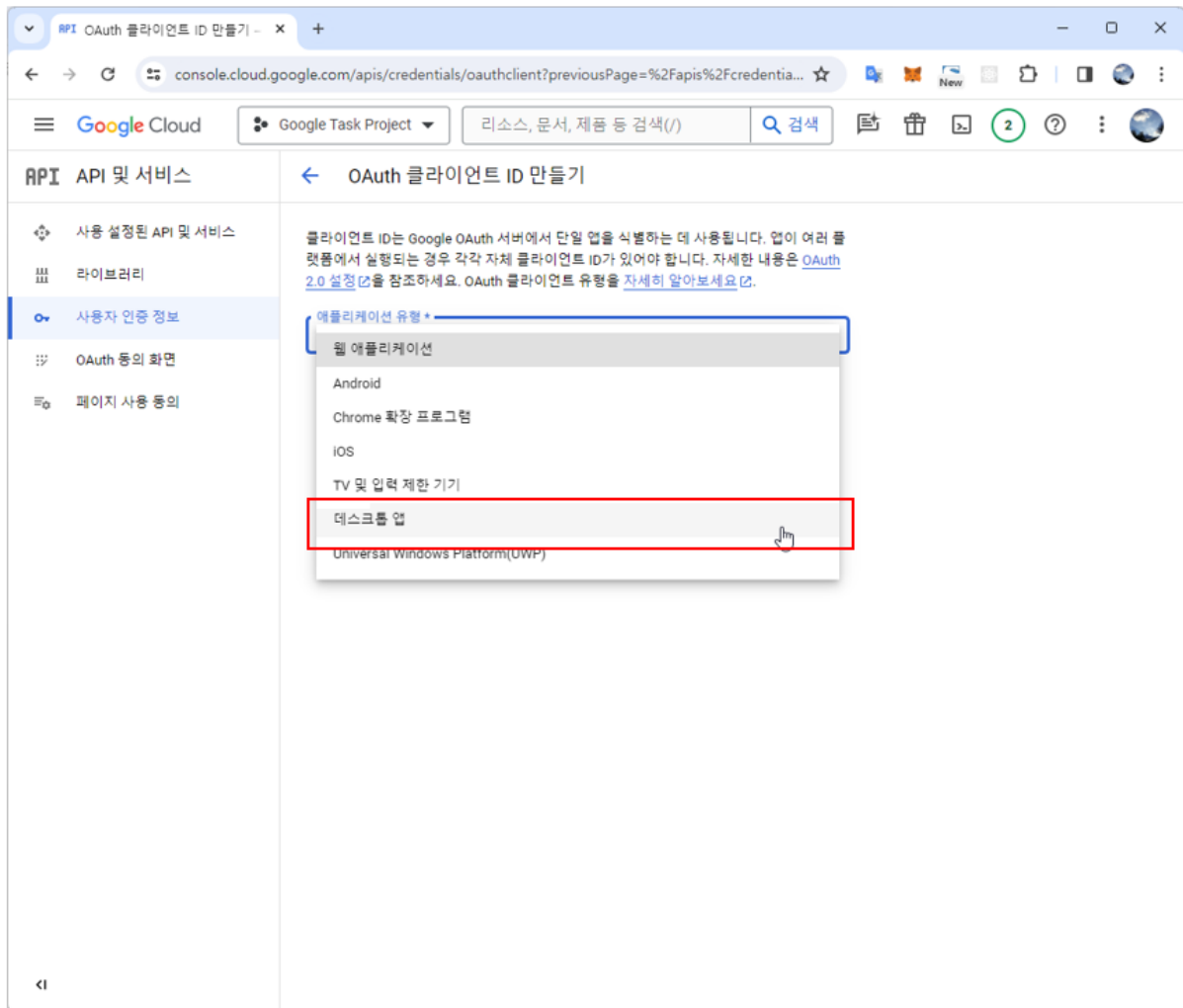
[그림 3-19] 사용자 인증 정보 메뉴 선택

[그림 3-19]와 같이 **사용자 인증 정보** 메뉴를 클릭합니다.



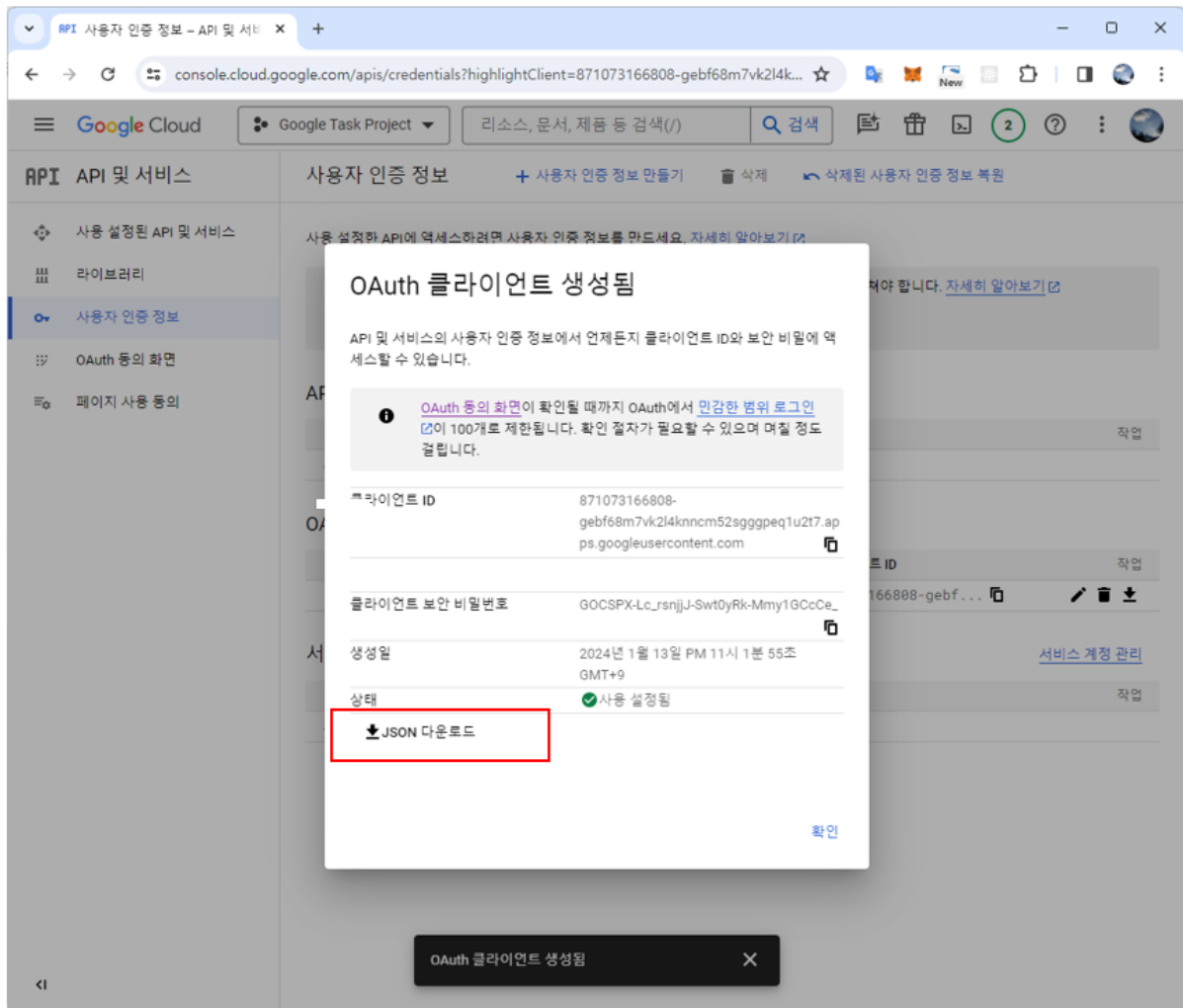
[그림 3-20] 사용자 인증 정보 만들기

[그림 3-20]의 (1)과 같이 **사용자 인증 정보 만들기** 를 클릭 한 후, (2)와 같이 **OAuth 클라이언트 ID** 를 선택합니다.



[그림 3-21] 애플리케이션 유형 선택

[그림 3-21]과 같이 애플리케이션 유형으로 **데스크톱 앱** 을 선택한 후, **만들기** 버튼을 클릭합니다.



[그림 3-22] OAuth 클라이언트 생성됨

[그림 3-22]와 같이 **JSON 다운로드** 버튼을 클릭해서 인증 정보가 담긴 파일을 다운로드 합니다.

다운로드 한 후에는 파일명을 **credentials.json**으로 변경해 둡니다.

4 Google Tasks API를 사용하기 위한 Gradle 프로젝트 설정

이제 Google Tasks API를 사용하기 위한 기본 설정이 끝났으니 IDE에서 Gradle 기반의 프로젝트를 생성한 후, Google Tasks API가 코드 레벨에서 정상 동작하는지 확인해 보겠습니다.

✔ Gradle 프로젝트 생성

Gradle 기반의 프로젝트를 생성하는 방법은 별도의 설명을 하지 않겠습니다.

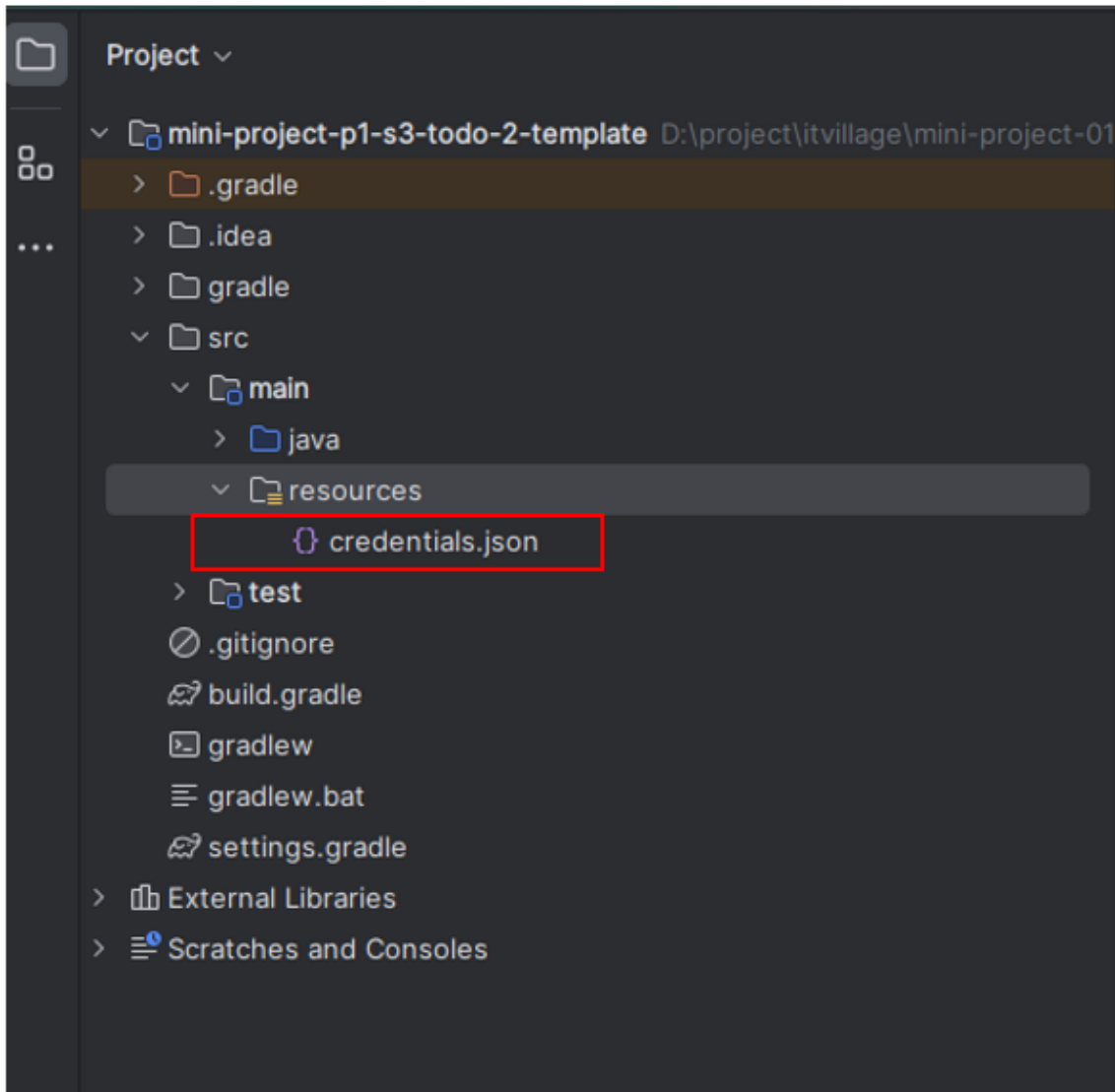
여러분들이 사용하고 있는 IntelliJ IDE에서 Gradle 기반의 프로젝트를 생성해 주세요.

✓ 의존 라이브러리 추가

생성한 프로젝트의 build.gradle 파일을 열어서 아래와 같이 Google Tasks API와 관련된 의존 라이브러리를 추가해 줍니다.

```
dependencies {  
    testImplementation platform('org.junit:junit-bom:5.9.1')  
    testImplementation 'org.junit.jupiter:junit-jupiter'  
    testImplementation 'org.hamcrest:hamcrest-core:2.2'  
    implementation 'com.google.api-client:google-api-client:2'  
    implementation 'com.google.oauth-client:google-oauth-clie'  
    implementation 'com.google.apis:google-api-services-tasks'  
}
```

✓ 사용자 인증 정보 json 파일 추가



[그림 4-1] 사용자 인증 정보 json 파일 추가

우리가 앞에서 다운로드 받아둔 `credentials.json` 파일을 여러분들이 생성한 프로젝트의 `src/main/resources/` 하위에 위치시킵니다.

✓ 샘플 클래스 생성

`src/main/java` 하위에 적절한 패키지를 생성 한 후, `TasksQuickstart` 클래스를 생성합니다.

✓ Google Tasks API 샘플 코드 추가하기

<https://developers.google.com/tasks/quickstart/java?hl=ko> 의 **샘플 설정** 항목에 있는 Google Tasks API의 샘플 코드 전체를 Copy 한 후, 조금 전에 생성한 **TasksQuickstart**에 붙여 넣기합니다.

```
package com.itvillage;

import com.google.api.client.auth.oauth2.Credential;
import com.google.api.client.extensions.java6.auth.oauth2.Auth;
import com.google.api.client.extensions.jetty.auth.oauth2.LocalizedReceiver;
import com.google.api.client.googleapis.auth.oauth2.GoogleAuth;
import com.google.api.client.googleapis.auth.oauth2.GoogleClientSecrets;
import com.google.api.client.googleapis.javanet.GoogleNetHttpTransport;
import com.google.api.client.http.javanet.NetHttpTransport;
import com.google.api.client.json.JsonFactory;
import com.google.api.client.json.gson.GsonFactory;
import com.google.api.client.util.store.FileDataStoreFactory;
import com.google.api.services.tasks.Tasks;
import com.google.api.services.tasks.TasksScopes;
import com.google.api.services.tasks.model.TaskList;
import com.google.api.services.tasks.model.TaskLists;
import java.io.FileNotFoundException;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.security.GeneralSecurityException;
import java.util.Collections;
import java.util.List;

public class TasksQuickstart {
    private static final String APPLICATION_NAME = "Google Ta
    private static final JsonFactory JSON_FACTORY = GsonFacto
    private static final String TOKENS_DIRECTORY_PATH = "toke

    /**
     * Global instance of the scopes required by this quickst
     * If modifying these scopes, delete your previously save
     */
    // (1) 사용 범위 설정
```

```

private static final List<String> SCOPES = Collections.synchronizedList(new ArrayList<>());
private static final String CREDENTIALS_FILE_PATH = "/credentials.json";

/**
 * Creates an authorized Credential object.
 *
 * @param HTTP_TRANSPORT The network HTTP Transport.
 * @return An authorized Credential object.
 * @throws IOException If the credentials.json file cannot be loaded.
 */
private static Credential getCredentials(final NetHttpTransport HTTP_TRANSPORT) throws IOException {
    // Load client secrets.
    InputStream in = TasksQuickstart.class.getResourceAsStream("credentials.json");
    if (in == null) {
        throw new FileNotFoundException("Resource not found: credentials.json");
    }
    GoogleClientSecrets clientSecrets =
        GoogleClientSecrets.load(JSON_FACTORY, new InputStreamFactory() {
            @Override public InputStream createInputStream() throws IOException {
                return in;
            }
        });

    // Build flow and trigger user authorization request.
    GoogleAuthorizationCodeFlow flow = new GoogleAuthorizationCodeFlow.Builder(
        HTTP_TRANSPORT, JSON_FACTORY, clientSecrets,
        new FileDataStoreFactory().setAccessType("offline")
    ).build();
    LocalServerReceiver receiver = new LocalServerReceiver.Builder().build();
    return new AuthorizationCodeInstalledApp(flow, receiver).authorize("user");
}

public static void main(String... args) throws IOException {
    // Build a new authorized API client service.
    final NetHttpTransport HTTP_TRANSPORT = GoogleNetHttpTransport.newTrustedTransport();
    Tasks service = new Tasks.Builder(HTTP_TRANSPORT, JSON_FACTORY,
        new ApplicationName("Tasks"))
        .build();

    // Print the first 10 task lists.
}

```

```

        TaskLists result = service.tasklists().list()
            .setMaxResults(10)
            .execute();
        List<TaskList> taskLists = result.getItems();
        if (taskLists == null || taskLists.isEmpty()) {
            System.out.println("No task lists found.");
        } else {
            System.out.println("Task lists:");
            for (TaskList tasklist : taskLists) {
                System.out.printf("%s (%s)\n", tasklist.getTi
            }
        }
    }
}

```

✓ Tasks API의 사용 범위(Scope) 변경

현재 여러분들이 복사 붙여 넣기한 샘플 코드는 Google Tasks API의 사용 범위가 read-only로 제한되어 있기 때문에 이 범위를 변경해 주어야 합니다.

위의 코드 4-1 샘플 코드에서 34번 째 라인 정도에 아래의 코드 라인을 찾으세요.

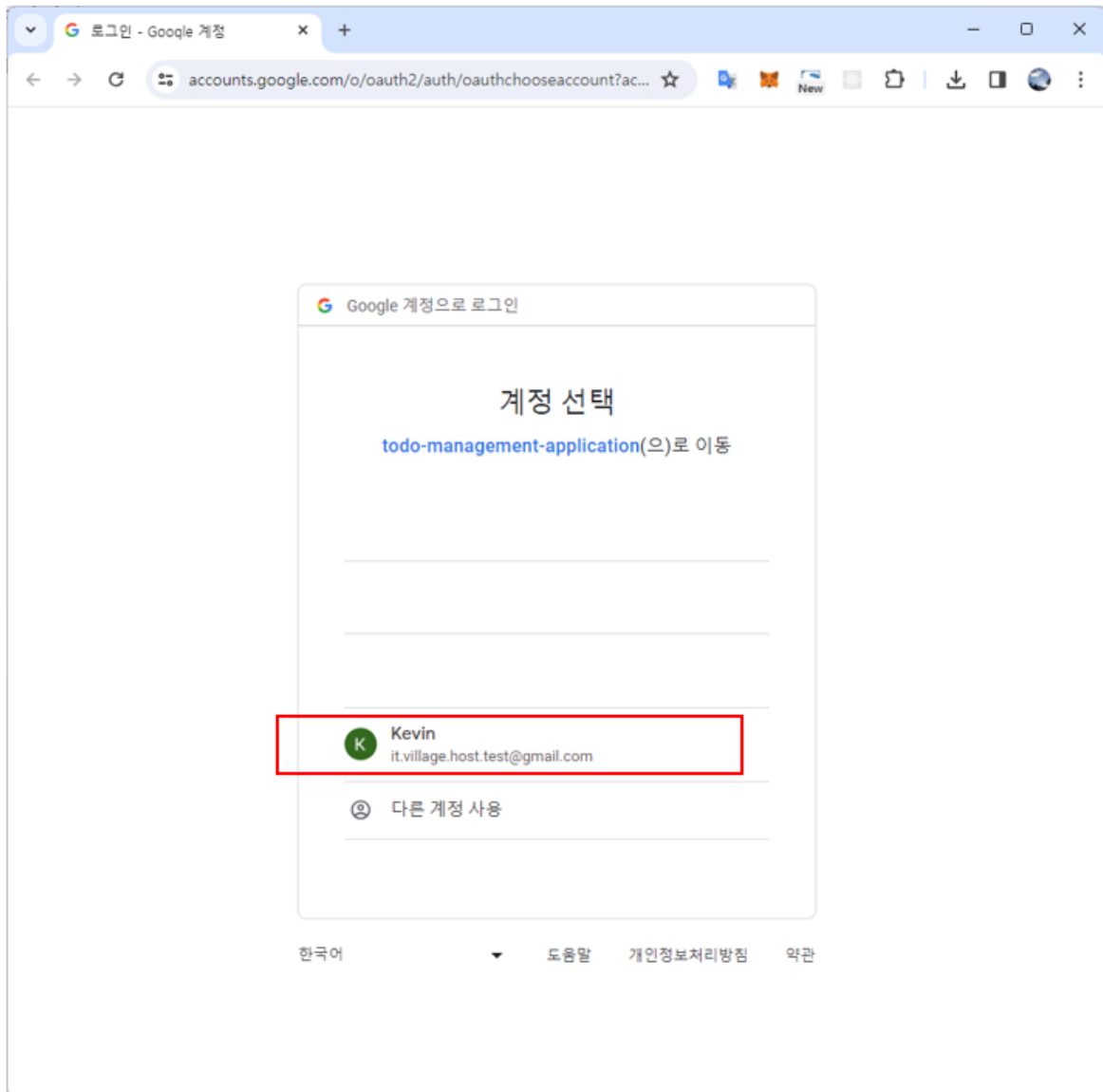
```
private static final List<String> SCOPES = Collections.single
```

여기서 `TasksScopes.TASKS_READONLY` 를 `TasksScopes.TASKS` 로 변경해 줍니다.

✓ TasksQuickstart 클래스 실행

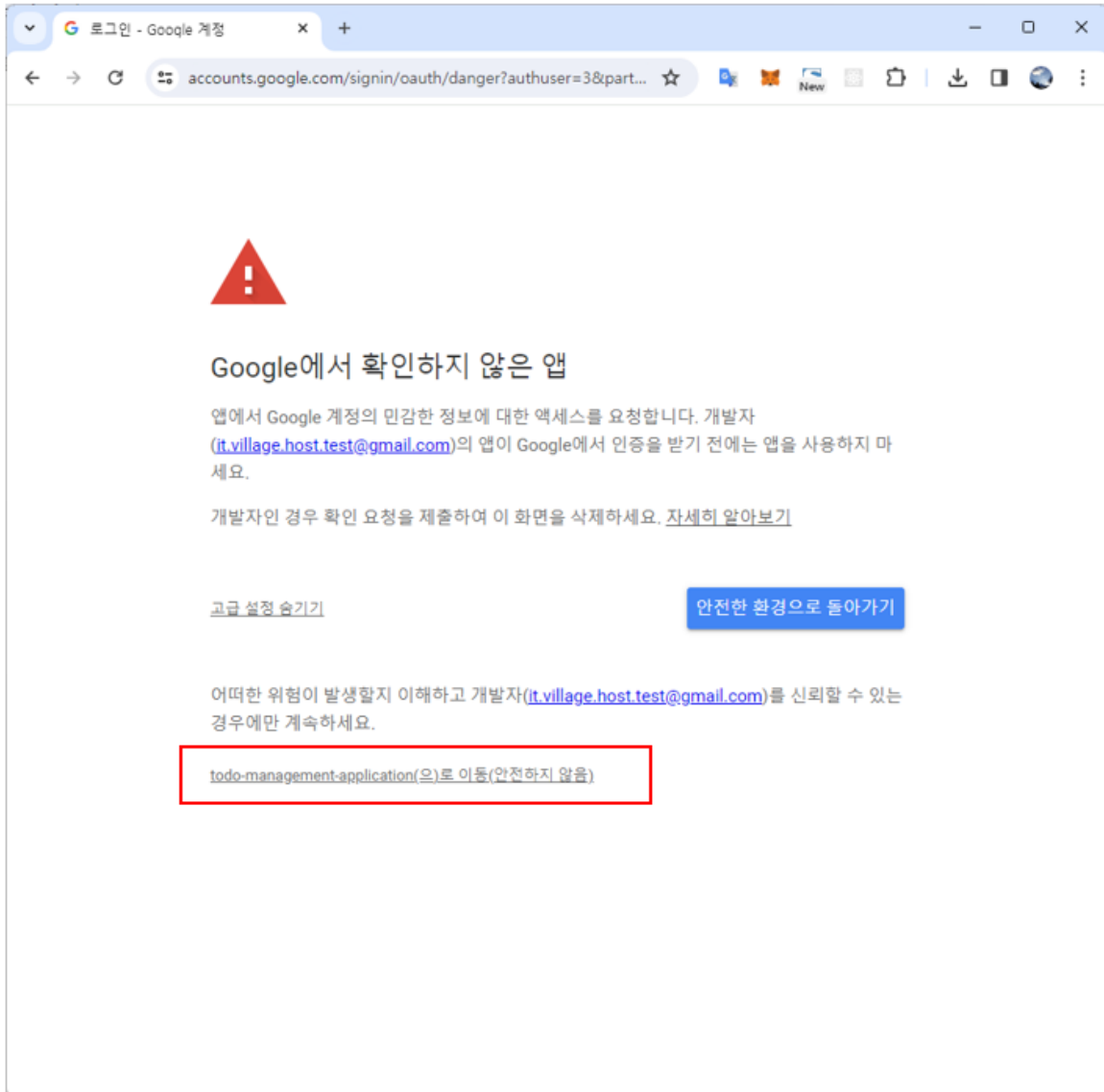
이제 IntelliJ IDE에서 TasksQuickstart 클래스를 실행 시킵니다.

애플리케이션이 정상적으로 실행되면 웹 브라우저에서 [그림 4-2]와 같이 `구글 계정으로 로그인` 화면이 보일겁니다.



[그림 4-2] 구글 계정으로 로그인 화면

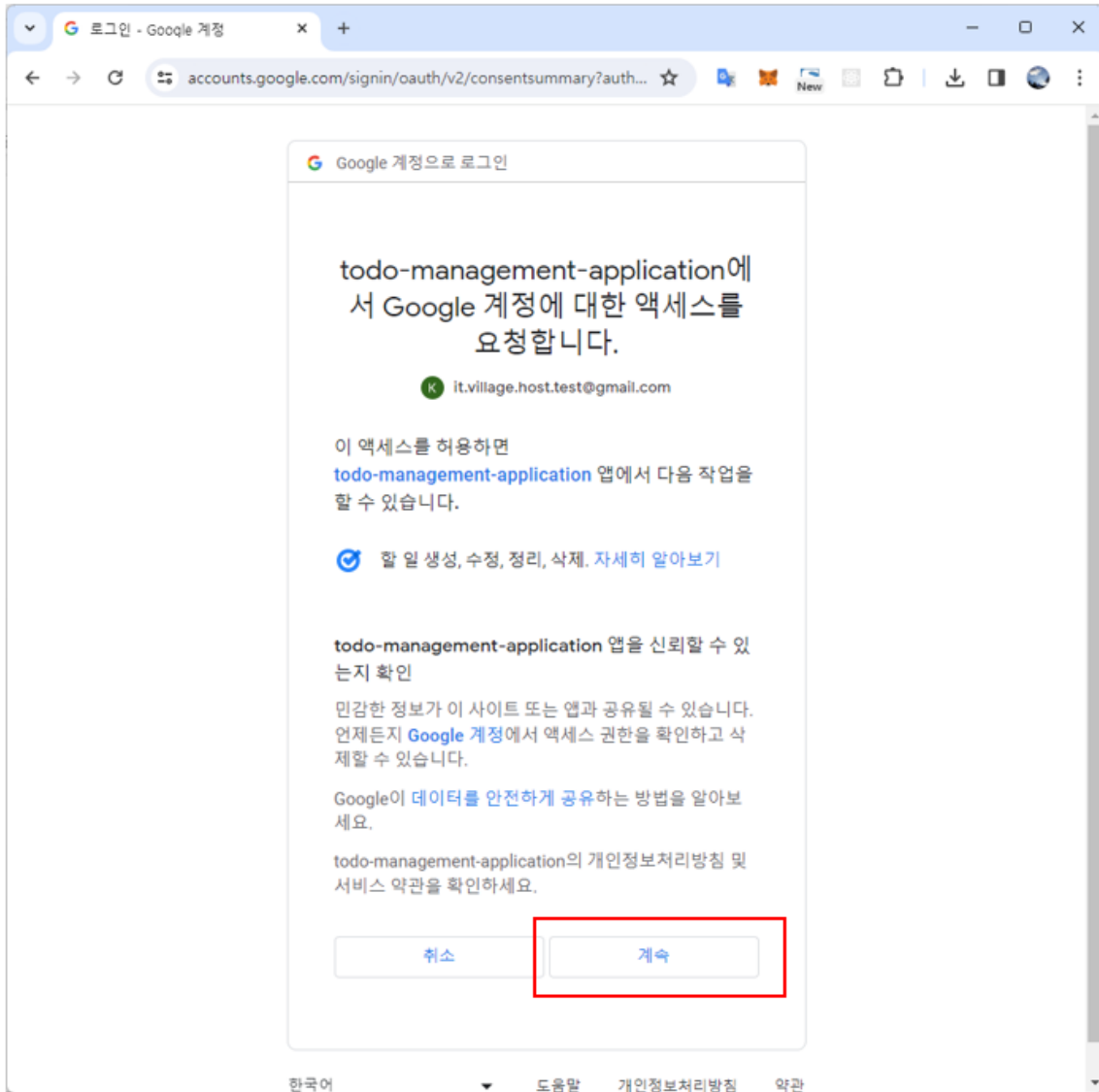
[그림 4-2]의 구글 계정으로 로그인 화면에서 여러분의 계정을 클릭합니다.



[그림 4-3] 구글 로그인

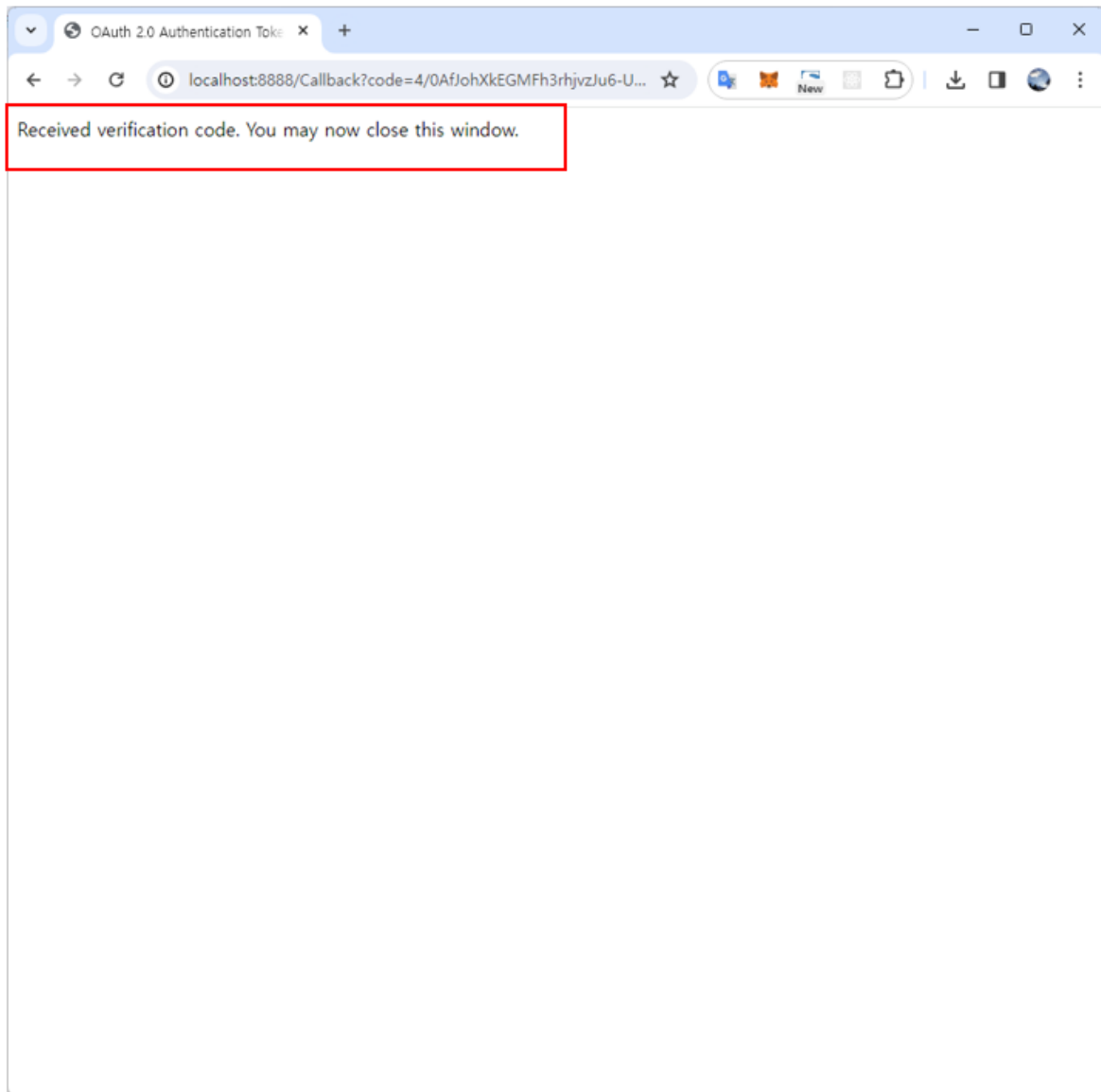
[그림 4-3]과 같이 **Google에서 확인하지 않은 앱**이라는 경고 화면이 뜨면 **고급**이라는 링크를 클릭하면, 아래 쪽에 여러분들이 Google Cloud Console에서 설정한 **xxxxxxxxx으로 이동** 링크가 보일 겁니다. 해당 링크를 클릭해서 로그인을 진행합니다.

💡 우리가 학습 목적으로 Google Cloud Console에서 구글에 별도의 인증 정보를 제출하지 않았기 때문에 [그림 4-3]과 같은 경고 화면이 뜬다는 사실을 기억하면 좋을 것 같습니다.



[그림 4-4] Google 계정에 대한 액세스 요청 화면

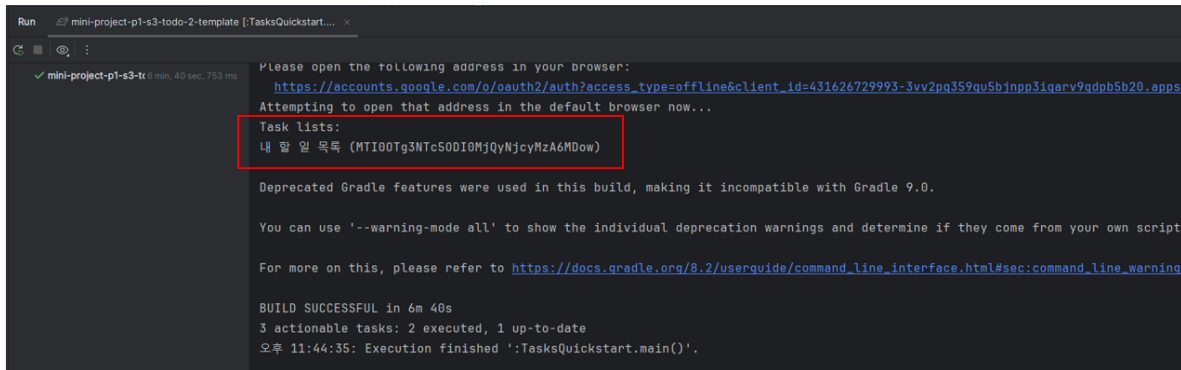
[그림 4-4]의 화면에서 **계속** 버튼을 클릭합니다.



[그림 4-5] 사용자 인증 성공 화면

[그림 4-5]와 같은 화면이 뜬다면 사용자 인증에 성공한 것입니다.

[그림 4-5]의 브라우저 창을 닫아 주세요.



```
Run mini-project-p1-s3-todo-2-template [TasksQuickstart... x]
mini-project-p1-s3-tr 6 min, 40 sec, 753 ms
Please open the following address in your browser:
https://accounts.google.com/o/oauth2/auth?access_type=offline&client_id=431626729993-3vv2pn359qu5bjnpp3iqarv9qddb5b20.apps
Attempting to open that address in the default browser now...
Task lists:
내 할 일 목록 (MTI00Tg3NTc5ODI0MjQyNjcyMzA6MDow)

Deprecated Gradle features were used in this build, making it incompatible with Gradle 9.0.

You can use '--warning-mode all' to show the individual deprecation warnings and determine if they come from your own script.

For more on this, please refer to https://docs.gradle.org/8.2/userguide/command_line_interface.html#sec:command_line_warnings

BUILD SUCCESSFUL in 6m 40s
3 actionable tasks: 2 executed, 1 up-to-date
오후 11:44:35: Execution finished 'TasksQuickstart.main()'
```

[그림 4-6] 샘플 코드 실행 결과

IntelliJ IDE로 돌아가서 실행 결과를 확인해 보면 여러분들이 사용할 기본 TaskList 정보를 조회한 후, 콘솔에 출력하고 있는 것을 볼 수 있습니다.

💡 구글로 로그인 사용자 인증 절차는 IntelliJ IDE에서 애플리케이션 최초 실행 시, 한번만 진행하면 되며, 이 후부터는 애플리케이션을 실행해도 별도의 인증 화면이 표시되지 않습니다.

만약 어떤 문제로 인해 구글로 로그인 인증 절차를 처음부터 다시 진행하고 싶다면 아래의 그림 4-6과 같이

`/tokens` 디렉토리 안의 인증 파일(`StoredCredential`)을 삭제한 후, 애플리케이션을 다시 실행하면 됩니다.

이제 Google Tasks API를 사용할 준비는 모두 끝났습니다.

이 후부터는 Google Tasks API를 사용해 할 일을 등록하고, 할 일을 조회한 후 완료 처리하는 로직을 구현하면 됩니다. ^^