

#1 Devise Omniauth, Heroku SSL, Bootstrap, Navbar, Footer

railsnew.io :

The screenshot shows a web browser window with the URL https://railsnew.io. The page title is "Rails is omakase". Below the title is a cartoon illustration of a chef's hat and a small Japanese restaurant icon. A text block reads: "But what if you have allergies 😱? (to a spring 🍽 in your soba ⚡, perhaps?) Fret not, purveyor of fine apps 🎉 for you are in good hands 🥰!" Below this is a subtext: "railsnew.io: the simplest way to generate a new Rails app with (or without) all the bells and whistles." At the bottom of the page, there is a dark input field with placeholder text "Enter the name of your awesome app!". Below the input field, the text "my_app" is typed. At the very bottom of the page, there is a command-line interface representation: "rails new my_app".

railsnew.io:

Ruby on Rails 애플리케이션을 새로 만들 때 필요한 옵션을 쉽게 선택할 수 있도록 도와주는 웹사이트입니다. 사용자는 이 사이트를 통해 새 Rails 애플리케이션의 초기 설정을 커스터마이즈할 수 있으며, 원하는 데이터베이스, 테스트 프레임워크, 자바스크립트 라이브러리 등을 선택한 후 생성할 애플리케이션의 구성 파일을 미리 볼 수 있습니다. (즉, 필요한 명령어가 뭔지 간략하게 보여줌)

```
require "active_support/core_ext/integer/time"  
Rails.application.configure do  
  config.hosts = nil  
  # Settings specified here will take precedence over those in config/application.rb.
```

development.rb파일의 "config.hosts = nil" 설정은 Ruby on Rails 애플리케이션에서 사용되며, 이 설정은 개발 모드에서 호스트 이름을 검증하는 기능을 비활성화합니다. Rails 6부터 도입된 config.hosts 설정은 웹 애플리케이션에 대한 요청이 허용된 호스트 목록에 있는 호스트 이름으로만 이루어지도록 제한하는 보안 기능입니다.

이 값은 nil로 설정하면 모든 호스트 이름에 대한 요청을 허용하게 되므로 개발 시 빠르고 유연하게 작업할 수 있지만, 보안상의 위험도 있을 수 있으므로 실제 운영 환경에서는 권장되지 않습니다.

```
ubuntu:~/environment/superails (main)$ rails g controller static_public landing_page privacy terms  
Running via Spring preloader in process 6097  
  create  app/controllers/static_public_controller.rb  
  route  get 'static_public/landing_page'  
get 'static_public/privacy'  
get 'static_public/terms'  
  invoke  erb  
  create  app/views/static_public  
  create  app/views/static_public/landing_page.html.erb  
  create  app/views/static_public/privacy.html.erb  
  create  app/views/static_public/terms.html.erb  
  invoke  test_unit  
  create  test/controllers/static_public_controller_test.rb  
  invoke  helper  
  create  app/helpers/static_public_helper.rb  
  invoke  test_unit  
  invoke  assets  
  invoke  scss  
    create  app/assets/stylesheets/static_public.scss  
ubuntu:~/environment/superails (main)$
```

그냥 저렇게 명령어를 실행하면 사용하지 않을 많은 파일들이 생성되어 버리므로 이를 방지해야함

git clean -fd와 git reset --hard는 두 개의 다른 Git 명령어로, 둘 다 워킹 디렉토리를 깨끗한 상태로 되돌리는 데 사용됩니다.

git clean -fd:

- f는 'force'의 약자로, 실제로 파일을 삭제하도록 Git에게 강제합니다.
- d는 디렉토리에도 적용하라는 의미로, Git에게 충격 되지 않는 파일뿐만 아니라 디렉토리도 삭제하도록 지시합니다.

git reset --hard:

- 현재 브랜치의 헤드가 가리키는 최근 커밋으로 모든 파일의 상태를 되돌립니다.
- 워킹 디렉토리와 인덱스(staging area)의 모든 변경 사항을 버리고 마지막 커밋 상태로 되돌립니다.

Write skinny scaffolds and generators

Mon, Nov 9, 2020 (week 46) • Yaroslav Shmarov



TLDR: add a few `--no-` tags to your rails generators to produce only the crucial files:

```
rails g controller home index --no-helper --no-assets --no-controller-specs --no-view-specs --no-test-framework
```

```
rails g scaffold product name description:text --no-helper --no-assets --no-controller-specs --no-view-specs --no-test-framework --no-jbuilder
```

Now, Let's dive in and make our rails generators cleaner!

Skinny Scaffold Generator

A usual scaffold like

```
rails g scaffold product name description:text
```

produces:

- Migration
- Model

```
ubuntu:~/environment/superails (main)$ rails g controller static_public landing_page privacy terms --no-helper --no-assets --no-controller-specs --no-view-specs --no-test-framework
Running via Spring preloader in process 6211
  create  app/controllers/static_public_controller.rb
  route  get 'static_public/landing_page'
get 'static_public/privacy'
get 'static_public/terms'
  invoke  erb
  create  app/views/static_public
  create  app/views/static_public/landing_page.html.erb
  create  app/views/static_public/privacy.html.erb
  create  app/views/static_public/terms.html.erb
ubuntu:~/environment/superails (main)$ █
```

빨간색 밑줄이 그어진 명령어 부분은 Ruby on Rails의 컨트롤러 생성 명령어입니다. 여기에서 `명령어가 사용되었습니다.` 이 명령어는 `static_public`이라는 이름의 컨트롤러와 연관된 뷰 페이지들(`landing_page`, `privacy`, `terms`)을 생성하는데 사용됩니다. 이때, 여러 플래그가 사용되어 헬퍼(helper), 자산(assets), 컨트롤러 스펙(controller specs), 뷰 스펙(view specs), 그리고 테스트 프레임워크(test framework) 파일들의 생성을 방지합니다.

플래그는 필요하지 않은 파일이 프로젝트에 추가되는 것을 방지하여 프로젝트를 더 깔끔하게 유지할 수 있도록 도와줍니다.

Routes

Routes match in priority from top to bottom

3개의 index페이지를 확인가능함

HTTP Verb Path

Path / Url	Path Match	Controller&Action
static_public_landing_page_path	GET /static_public/landing_page(.:format)	static_public#landing
static_public_privacy_path	GET /static_public/privacy(.:format)	static_public#privacy
static_public_terms_path	GET /static_public/terms(.:format)	static_public#terms

StaticPublic#landing_page

Find me in app/views/static_public/landing_page.html.erb

StaticPublic#privacy

Find me in app/views/static_public/privacy.html.erb

```
run Tools Window Support Preview Run
puma - "ip-172-31-4-66" x development.rb x routes.rb • +
```

```
1 Rails.application.routes.draw do
2   root 'static_public#landing_page'
3   get 'static_public/landing_page'
4   get 'static_public/privacy'
5   get 'static_public/terms'
6   # For details on the DSL available within this file, see https://guides.rubyonrails.org/routing.html
7 end
```

지금상태로 rails s 를 실행해보면
static_public/privacy 로 route가 형성됨
/privacy가 되게끔 routes.rb수정

Amazonaws.com/static_public/privacy

```
puma - "ip-172-31-4-66" x development.rb x routes.rb
```

```
1 Rails.application.routes.draw do
2   root 'static_public#landing_page'
3   # get 'static_public/landing_page'
4   get 'privacy', to: 'static_public#privacy'
5   get 'static_public/terms'
6   # For details on the DSL available within this file, see https://guides.rubyonrails.org/routing.html
7 end
```

Path / Url	Path Match	Controller&Action
root_path	GET /	static_public#landing_page
privacy_path	GET /privacy(.:format)	static_public#privacy
static_public_terms_path	GET /static_public/terms(.:format)	static_public#terms

```
puma - "ip-172-31-4-66" x development.rb x routes.rb
```

```
1 Rails.application.routes.draw do
2   root 'static_public#landing_page'
3   get 'privacy', to: 'static_public#privacy'
4   get 'terms', to: 'static_public#terms'
5   # For details on the DSL available within this file, see https://guides.rubyonrails.org/routing.html
6 end
```

Path / Url	Path Match	Controller&Action
rails_postmark_inbound_emails_path	POST /rails/action_mailbox/postmark/inbound_emails(.:format)	action_mailbox#inbound_emails#create
rails_update_inbound_emails_path	PUT /rails/action_mailbox/inbound_emails(.:format)	action_mailbox#inbound_emails#update

heroku에 push하기 (9:36~)

git remote -v : 명령어는 Git에서 현재 프로젝트에 등록된 모든 원격 저장소(remote repositories)의 URL을 보여줍니다. -v 또는 --verbose 옵션은 '자세히'라는 의미로, 각 원격 저장소의 이름과 함께 가져오기(fetch)와 일어넣기(push)를 위한 URL을 나타냅니다. 이 정보는 원격 저장소와의 상호작용을 할 때 어떤 URL을 참조해야 하는지 확인하는데 유용합니다. 예를 들어, GitHub에 호스팅된 프로젝트의 원격 저장소 주소를 확인할 때 자주 사용됩니다.

```
node - "ip-172-31-4-66" x development.rb x routes.rb x +
```

```
ubuntu:~/environment/superails (main)$ git status
ubuntu:~/environment/superails (main)$ git remote -v
origin git@github.com:yshmarov/superails.git (fetch)
origin git@github.com:yshmarov/superails.git (push)
ubuntu:~/environment/superails (main)$ heroku create
  Warning: heroku update available from 7.52.0 to 7.54.1.
Creating app... done, ⚡ mighty-savannah-88659
https://mighty-savannah-88659.herokuapp.com/ | https://git.heroku.com/mighty-savannah-88659.git
ubuntu:~/environment/superails (main)$ heroku update
  Warning: update with: snap refresh heroku
heroku: Updating CLI... not updatable
```

heroku create:

Heroku에 새로운 애플리케이션을 생성합니다. 이 명령을 실행하면 Heroku에서 애플리케이션에 대한 URL을 제공하고, 이 URL을 통해 배포된 애플리케이션에 접근할 수 있게 됩니다.

heroku update:

Heroku CLI(Command Line Interface)를 최신 버전으로 업데이트합니다. 다만, 스크립트에는 'update with: snap refresh heroku'라는 메시지가 보이는 데, 이는 Heroku CLI가 snap 패키지 매니저를 통해 설치되었을 때 해당 방법으로 업데이트해야 힘을 의미합니다.

```

bash - "ip-172-31-4-66" x development.rb x ro
ubuntu:~/environment/superails (main) $ git status
ubuntu:~/environment/superails (main) $ git remote -v
origin git@github.com:yshmarov/superails.git (fetch)
origin git@github.com:yshmarov/superails.git (push)
ubuntu:~/environment/superails (main) $ heroku create
-> Warning: heroku update available from 7.52.0 to 7.54.1.
Creating app... done, ⚡ mighty-savannah-08659
https://mighty-savannah-08659.herokuapp.com/ | https://git.heroku
ubuntu:~/environment/superails (main) $ heroku update
-> Warning: update with: snap refresh heroku
heroku: Updating CLI... not updatable
Updating completions... done
ubuntu:~/environment/superails (main) $ heroku rename superails
-> Warning: heroku update available from 7.52.0 to 7.54.1.
Renaming mighty-savannah-08659 to superails... done
https://superails.herokuapp.com/ | https://git.heroku.com/superai
-> Don't forget to update git remotes for all other local check
Git remote heroku updated
ubuntu:~/environment/superails (main) $ git remote -v
heroku https://git.heroku.com/superails.git (fetch)
heroku https://git.heroku.com/superails.git (push)
origin git@github.com:yshmarov/superails.git (fetch)
origin git@github.com:yshmarov/superails.git (push)
ubuntu:~/environment/superails (main) $ 

```

Salesforce Platform

HEROKU

Jump to Favorites, Apps, Pipelines, Spaces...

Jun 5 at 12:04 PM

askdemos
collegecrm
doestmystartupeasuck
edurge
gorocrm
intellilex
saasblog
shopify
superails
superauth

heroku.com Blogs Careers Documentation Support

Salesforce Platform

HEROKU

Jump to Favorites, Apps, Pipelines, Spaces...

Personal > superails

Overview Resources Deploy Metrics Activity Access Settings

Installed add-ons \$0.00/month Configure Add-ons Latest activity

There are no add-ons for this app You can add add-ons to this app and they will show here. [Learn more](#)

Dyno formation \$0.00/month Configure Dynos

This app has no process types yet Add a Procfile to your app in order to define its process types. [Learn more](#)

Collaborator activity

Manage Access

There is no recent activity on this app Collaborator activity will be shown when there are recent deploys

Salesforce Platform

HEROKU

Jump to Favorites, Apps, Pipelines, Spaces...

Add this app to a pipeline

Create a new pipeline or choose an existing one and add this app to a stage in it.

Add this app to a stage in a pipeline to enable additional features

Pipelines let you connect multiple apps together and promote code between them. [Learn more](#)

Pipelines connected to GitHub can enable review apps, and create apps from GitHub. [Learn more](#)

Choose a pipeline



Deployment method

Heroku Git Use Heroku CLI GitHub Connect to GitHub Container Registry Use Heroku CLI

Deploy using Heroku Git

Use git in the command line or a GUI tool to deploy this app.

Install the Heroku CLI

Download and install the [Heroku CLI](#). If you haven't already, log in to your Heroku account and follow the prompts to create a new SSH public key.

```
$ heroku login
```

Create a new Git repository

Initialize a git repository in a new or existing directory

```
$ cd my-project/
$ git init
$ heroku git:remote -a superails
```

Deploy your application

Deployment method

Heroku Git Use Heroku CLI GitHub Connect to GitHub Container Registry Use Heroku CLI

Connect to GitHub

Connect this app to GitHub to enable code diffs and deploys.

Search for a repository to connect to

yshmarov Repo name Search

Missing a GitHub organization? Ensure Heroku Dashboard has team access

Automatic deployments
Enables a chosen branch to be automatically deployed to this app.

You can now change your main deploy branch from "master" to "main" for both manual and automatic deployments. [Follow the instructions here.](#)

Enable automatic deployments from GitHub

Every push to the branch you specify here will deploy a new version of this app. Deployments happen automatically; be sure that this branch is always in a deployable state and any tests have passed before you push. [Learn more.](#)

Choose a branch to deploy

main

Wait for CI to pass before deploy

Only enable this option if you have a Continuous Integration service configured on your repo.

[Enable Automatic Deployments](#)



git push heroku main 명령어 실행

Manual deploy

Deploy the current state of a branch to this app.

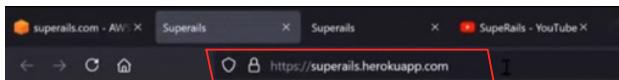
Deploy a GitHub branch

This will deploy the current state of the branch you specify below. [Learn more.](#)

Choose a branch to deploy

main

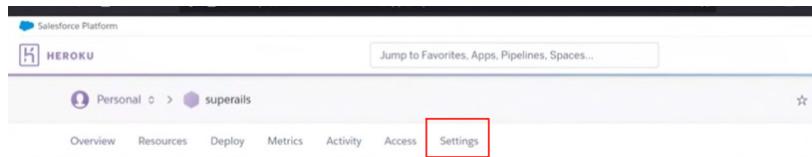
[Deploy Branch](#)



StaticPublic#landing_page

Find me in app/views/static_public/landing_page.html.erb

도메인 이름 추가하기:



Domains

You can add custom domains to any Heroku app, then visit [Configuring DNS](#) to setup your DNS target.

Your app can be found at <https://superails.herokuapp.com/>

[Add domain](#)

Filter domains

Custom domains will appear here

Custom domains allow you to access your app via one or more non-Heroku domain names (for example, www.yourcustomdomain.com)

Transfer Ownership

Transfer this app to your personal account or a team you are a member of. [Learn more](#)

Choose app owner

yshmarov@gmail.com

[Transfer app...](#)

Automated Certificate Management (ACM) is available for applications running on paid dynos to automate your SSL security.

Certificates acquired elsewhere may be configured using the [Manual Certificate](#) option.

Domains

You can add custom domains to any Heroku app, then visit [Configuring DNS](#) to setup your DNS target.

Your app can be found at <https://superails.herokuapp.com/>

Filter domains

Domain name: superails.com

SSL Certificate: Please enter a domain or select one from your existing certs.

No SNI Endpoint

See the [Rules on adding domains](#) section if you receive the error message "example.com is currently in use by another app".

Automated Certificate Management (ACM) is available for applications running on paid dynos to automate your SSL security.

Certificates acquired elsewhere may be configured using the [Manual Certificate](#) option.

Domains

You can add custom domains to any Heroku app, then visit [Configuring DNS](#) to setup your DNS target.

Your app can be found at <http://superails.com>

Filter domains

Domain Name: superails.com
DNS Target: guarded-mule-66ft0tdwndcjskjej

Transfer Ownership

Transfer this app to your personal account or a team you are a member of. [Learn more](#)

Choose app owner: yshmarov@gmail.com

Transfer app...

Cancel Next

Automated Certificate Management (ACM) is available for applications running on paid dynos to automate your SSL security.

Certificates acquired elsewhere may be configured using the [Manual Certificate](#) option.

Domains

You can add custom domains to any Heroku app, then visit [Configuring DNS](#) to setup your DNS target.

You have multiple custom domains enabled

Add domain

Filter domains

Domain Name: superails.com
DNS Target: guarded-mule-66ft0tdwndcjskjejh1hh2g5...

Domain Name: www.superails.com
DNS Target: secure-walrus-85dvjauw2ojlutr16fg36mz.h...

Domain Name: secure-walrus-85dvjauw2ojlutr16fg36mz.h...

dns target을 domain provider에 추가하기:

The screenshot shows the Namecheap domain management interface. On the left sidebar, there are links for Dashboard, Expiring / Expired, Domain List, Hosting List, Private Email, SSL Certificates, Apps, and Profile. The main content area displays a list of domains under 'Recommended for you'. One domain, 'superalrs.com', has its 'Manage' button highlighted with a red box. Below the list is a message 'Need help? We're always here for you.' and a 'Chat with a Live Person' button.

This screenshot shows the 'Details' page for the 'superalrs.com' domain. The top navigation bar includes Domains, Hosting, WordPress, Email, Apps, Security, Transfer To Us, Help Center, and Account. The 'Advanced DNS' tab is selected. A red box highlights the 'HOST-RECORDS' section, which lists two URL Redirect Records:

Type	Host	Value	TTL
URL Redirect Record	@	https://www.youtube.com/ch...	Unmasked
URL Redirect Record	www	https://www.youtube.com/ch...	Unmasked

This screenshot shows the 'Advanced DNS' record creation form. A red box highlights the 'ADD NEW RECORD' button and the input fields for Type, Host, Value, and TTL. A CNAME Record is being added with the host '@' and value 'guarded-mule-66f07dwndcjskkeh1hh2g5 herokudns.com'. The TTL is set to 'Automatic'.

The screenshot displays a DNS management interface with a sidebar containing links like Dashboard, Expiring / Expired, Domain List, Hosting List, Private Email, SSL Certificates, Apps, and Profile. The main area has tabs for DNS TEMPLATES, Choose DNS Template, HOST RECORDS, and Advanced DNS. Under HOST RECORDS, there is a table with columns Type, Host, Value, and TTL. Two CNAME records are listed: one for '@' pointing to 'guarded-mule-66f0tdwndcjkkeh1hh2g5.herokuapp.com' with TTL Automatic, and another for 'www' pointing to 'secure-walrus-85dyaauw2ojuirif6g56mz.herokuapp.com' with TTL Automatic. An 'ADD NEW RECORD' button is at the bottom. Below this, sections for DNSSEC and MAIL SETTINGS are shown.

ACM satatus를 변경(연결을 보안연결로 변경? 13:00~) https로 사용가능

15:55 bootstrap 스타일링 추가 : <https://blog.corsego.com/rails-bootstrap-5-yarn>

`yarn add bootstrap` 명령어는 Windows 환경에서도 동작합니다. Yarn은 크로스 플랫폼 패키지 매니저이므로 Windows, macOS, Linux 등 다양한 운영 체제에서 사용할 수 있습니다.

이 명령어는 다음과 같은 기능을 수행합니다:

1. Yarn을 사용하여 Bootstrap 라이브러리를 현재 프로젝트의 종속성(dependency)으로 추가합니다.
2. 프로젝트의 `package.json` 파일에 Bootstrap을 추가하며, `yarn.lock` 파일을 업데이트하여 설치된 버전의 정보를 기록합니다.
3. Bootstrap 관련 파일들이 프로젝트의 `node_modules` 디렉토리에 설치됩니다.

이를 통해 프로젝트에서 Bootstrap의 CSS 및 JavaScript 구성요소를 사용할 수 있게 됩니다.

CDN 링크를 사용하는 방식과 패키지 매니저(yarn)를 통해 설치하는 방식에는 몇 가지 차이가 있습니다:

1. CDN (Content Delivery Network):
 - 속도와 효율성: 사용자는 여러 지역에 분산된 서버로부터 파일을 받기 때문에 더 빠른 로딩 속도를 경험할 수 있습니다.
 - 캐싱의 이점: 사용자가 다른 웹사이트에서 같은 Bootstrap CDN을 사용했을 경우, 브라우저 캐시를 통해 더 빠르게 로드할 수 있습니다.
 - 간편성: HTML 파일에 링크를 추가하는 것만으로도 즉시 사용할 수 있어서 간단합니다.
 - 버전 관리: 특정 버전을 명시적으로 지정해야 하며, 업데이트를 위해서는 HTML 코드를 변경해야 합니다.
2. 패키지 매니저 (Yarn 또는 npm):
 - 프로젝트 내 종속성 관리: 프로젝트의 `package.json`과 `yarn.lock` 파일을 통해 관리되므로, 팀원 간 버전 호환성을 유지하기 쉽습니다.
 - 커스터마이징: 필요한 Bootstrap SCSS 파일만 선택적으로 가져와서 커스텀 테마나 스타일을 적용하기 용이합니다.
 - 빌드 도구와의 통합: Webpack이나 Gulp 같은 빌드 도구와 통합하여 사용하기 용이합니다.
 - 오프라인 작업: 인터넷 연결이 끊겼을 때도 로컬에 설치된 패키지를 사용할 수 있습니다.
 - 업데이트 관리: 패키지 매니저를 통해 의존성을 업데이트하는 것이 더 쉽습니다.

The screenshot shows a browser window with multiple tabs. On the left, there's a sidebar for 'CSS Reference' with categories like 'CSS Colors', 'CSS Color Values', 'CSS Default Values', and 'CSS Entities'. The main content area displays a grid of color swatches with names like AliceBlue, AntiqueWhite, Aqua, Aquamarine, Azure, Beige, Bisque, Black, BlanchedAlmond, Blue, BlueViolet, and Brown. A red box highlights the 'AntiqueWhite' swatch. To the right, a code editor shows a file named 'application.js' with the following code:

```

1 // Import "bootstrap";
2
3 body { background-color: AntiqueWhite; }
4

```

Home Privacy Terms

StaticPublic#landing_page

Find me in app/views/static_public/landing_page.html.erb

body tag에 background-color 설정

StaticPublic#landing_page

Find me in app/views/static_public/landing_page.html.erb

```

<body class="container">
  <% link_to 'Terms', root_path %>
  <% link_to 'Privacy', privacy_path %>
  <% link_to 'Terms', terms_path %>
</body>
</html>

```

```

1 @import "bootstrap";
2
3 body { background-color: AntiqueWhite; }
4
5 a { text-decoration: none; }
6

```

Bootstrap의 .container 클래스는 반응형 고정 너비 컨테이너를 만드는 데 사용됩니다. 이 클래스는 요소의 최대 너비를 설정하고 자동으로 양쪽에 여백을 주어 컨텐츠가 중앙에 위치하도록 합니다. 또한, 반응형 레이아웃을 위해 다양한 뷰포트 크기에서 최대 너비가 조정됩니다. 예를 들어, 작은 화면에서는 컨테이너가 더 좁게 설정되고, 큰 화면에서는 더 넓게 설정되어 적절한 레이아웃을 유지합니다.

Automatic deploys

Enables a chosen branch to be automatically deployed to this app.

You can now change your main deploy branch from "master" to "main" for both manual and automatic deploys, please follow the instructions [here](#).

Automatic deploys from `main` are enabled

Every push to `main` will deploy a new version of this app. **Deploys happen automatically**: be sure that this branch in GitHub is always in a deployable state and any tests have passed before you push. [Learn more](#).

Wait for CI to pass before deploy

Only enable this option if you have a Continuous Integration service configured on your repo.

[Disable Automatic Deploys](#)

Manual deploy

Deploy the current state of a branch to this app.

Deploy a GitHub branch

This will deploy the current state of the branch you specify below. [Learn more](#).

Choose a branch to deploy

`main`

Automatic deploy (자동 배포) 기능은 Heroku 같은 클라우드 플랫폼에서 제공하는 기능으로, 연결된 Git 저장소의 특정 브랜치에 새로운 커밋이 푸시될 때마다 애플리케이션을 자동으로 배포합니다. 이 기능을 활성화하면 개발자가 수동으로 배포 버튼을 클릭할 필요 없이 최신 코드 변경 사항을 즉시 배포할 수 있습니다.



main	branch	tags	Go to file	Add file	Code
yshmarov add bootstrap			2f2bc49 now	3 commits	
app	add bootstrap			now	
bin	initialize app			22 minutes ago	
config	static public controller with pages			17 minutes ago	
db	initialize app			22 minutes ago	
lib	initialize app			22 minutes ago	

About

No description, website, or topics provided.

[Readme](#)

Releases

No releases published
[Create a new release](#)

production link
를 배치할 수 있음

The screenshot shows the GitHub repository settings for 'superails'. A red box highlights the 'Website' input field, which contains the URL <https://yshmarov.github.io/superails/>. Below it, the 'Topics' section is expanded, showing options like 'Releases', 'Packages', and 'Environments', with 'Releases' checked. The 'About' section is also visible on the right.

```

superails.com - /home/u
superails
> git
> app
> bin
> config
> environments
  > development.rb
  > production.rb
  > test.rb
> initializers
> locales
> webpack
  application.rb
  boot.rb
  cable.yml
  credentials.yml.enc
  database.yml
  environment.rb
  master.key
  puma.rb
  routes.rb

```

```

# Enable serving of images, stylesheets, and JavaScripts from an asset
# config.asset_host = 'http://assets.example.com'

# Specifies the header that your server uses for sending files.
# config.action_dispatch.x_sendfile_header = 'X-Sendfile' # for Apache
# config.action_dispatch.x_sendfile_header = 'X-Accel-Redirect' # for NGINX

# Store uploaded files on the local file system (see config/storage.yml for options).
config.active_storage.service = :local

# Mount Action Cable outside main process or domain.
# config.action_cable.mount_path = nil
# config.action_cable.url = "ws://example.com/cable"
# config.action_cable.allowed_request_origins = [ 'http://example.com', /http:\/\/\w{3}.example./ ]

# Force all access to the app over SSL, use Strict-Transport-Security, and use secure cookies.
config.force_ssl = true

# Include generic and useful information about system operation, but avoid logging too much
# information to avoid inadvertent exposure of personally identifiable information (PII).
config.log_level = :info

# Prepend all log lines with the following tags.
# config.log_tags = [ :request_id ]

# Use a different cache store in production.
# config.cache_store = :mem_cache_store

# Use a real queuing backend for Active Job (and separate queues per environment).
# config.active_job.queue_adapter = :resque
# config.active_job.queue_name_prefix = "superails_production"

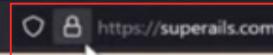
```

environments/production.rb에서
config.force_ssl = true 설정

config.force_ssl = true 설정은 Ruby on Rails 애플리케이션에서 환경 설정 파일인 environments/production.rb에 추가할 수 있습니다.

이 설정은 애플리케이션을 운영 환경에서 실행할 때 모든 트래픽을 HTTPS를 통해서만 받도록 강제합니다. 이는 데이터 전송 중 정보가 암호화되어 안전하게 처리됨을 보장하고, Man-in-the-Middle (MitM) 공격과 같은 보안 위협으로부터 사용자의 데이터를 보호하는 데 도움을 줍니다.

따라서, 이 설정은 프로덕션 환경에서 사용자의 세션과 쿠키가 암호화되어 전송되도록 하여 보안을 강화하는 중요한 기능을 수행합니다.



StaticPublic#landing_page

Find me in app/views/static_public/landing_page.html.erb

기초적인 스타일링 : FontAwesome with Yarn and Webpacker in rails

참조 : <https://blog.corsego.com/fontawesome-importmaps-rails7>

참조 : <https://blog.corsego.com/rails-6-install-fontawesome-with-webpacker>

```
ubuntu:~/environment/superrails (main) $ ^C  
ubuntu:~/environment/superrails (main) $ yarn add @fortawesome/fontawesome-free
```

Webpacker in Rails 6?

Mon, Oct 26, 2020 · Yaroslav Shmarov

Using FontAwesome yarn package manager

1. console:

```
yarn add @fortawesome/fontawesome-free
```

2. javascript/packs/application.js:

```
import "@fortawesome/fontawesome-free/css/all"
```

3. Check if it works:

Add couple of icons in any .html.erb (view) file:

```
<i class="far fa-address-book"></i>  
<i class="fab fa-github"></i>
```

Tips and tricks

- Use smth like `fa-3x` for font size.
- Use `fa-spin` to make any icon spin. [Animating Icons](#)
- Use it in a link with a block

```
<%= link_to root_path do %>  
<i class="far fa-gem fa-spin fa-3x"></i>  
Home  
<% end %>
```

That's it! 😊

fontAwesome에서 원하는 아이콘 선택해서 사용

The screenshot shows the Font Awesome v6 Beta website. At the top, there's a banner with the text "Ready to pour... the Font Awesome 6 Beta!" and a message about the beta release. Below the banner is a search bar with the placeholder "Search 1,609 icons for...". To the left of the search bar is a sidebar with category filters: "Free" (checked), "Pro Only", "Solid", "Regular", "Light", "Duotone", "Brands", "Latest Release", "Accessibility", "Alert", "Animals", and "Arrows". On the right side, there's a grid of 1,609 icons labeled "All 1,609 Awesome Icons". Each icon has a small preview and a corresponding name below it. In the bottom right corner of the grid, there's a video player showing a person speaking. The bottom of the page shows a terminal window with some code and a screenshot of a browser displaying the result.

Home Privacy Terms

StaticPublic#landing_page

Find me in app/views/static_public/landing_page.html.erb



devise gem 을 이용한 로그인 기능 구현
참조 : <https://github.com/heartcombo/devise>



Devise is a flexible authentication solution for Rails based on Warden. It:

- Is Rack based;
- Is a complete MVC solution based on Rails engines;
- Allows you to have multiple models signed in at the same time;
- Is based on a modularity concept: use only what you really need.

It's composed of 10 modules:

- Database Authenticatable: hashes and stores a password in the database to validate the authenticity of a user while signing in. The authentication can be done both through POST requests or HTTP Basic Authentication.
- Omniauthable: adds OmniAuth (<https://github.com/omniauth/omniauth>) support.
- Confirmable: sends emails with confirmation instructions and verifies whether an account is already confirmed during sign in.
- Recoverable: resets the user password and sends reset instructions.
- Registerable: handles signing up users through a registration process, also allowing them to edit and destroy their account.
- Rememberable: manages generating and clearing a token for remembering the user from a saved cookie.
- Trackable: tracks sign in count, timestamps and IP address.
- Timeatable: expires sessions that have not been active in a specified period of time.
- Validatable: provides validations of email and password. It's optional and can be customized, so you're able to define your own validations.
- Lockable: locks an account after a specified number of failed sign-in attempts. Can unlock via email or after a specified time period.

rails g devise:install 실행하여 console에 나타나는 지시사항들 실행하기

```
bash - "ip-172-31-4-66" ✘ Gemfile
* Ensure you have devise installed in your environment. If not, run `gem install devise`.
* An example of default_url_options appropriate for a development environment
in config/environments/development.rb:
config.action_mailer.default_url_options = { host: 'localhost', port: 3000 }

In production, :host should be set to the actual host of your application.

* Required for all applications. *

2. Ensure you have defined root_url to *something* in your config/routes.rb.
For example:
root to: "home#index"

* Not required for API-only Applications *

3. Ensure you have flash messages in app/views/layouts/application.html.erb.
For example:
<p class="notice"><%= notice %></p>
<p class="alert"><%= alert %></p>

* Not required for API-only Applications *

4. You can copy Devise views (for customization) to your app by running:
rails g devise:views

* Not required *
```

rails generate devise User 실행하여 model 생성 후,
rails db:migrate 실행하여 생성된 마이그레이션 파일들을 스키마에 적용하기

```
ubuntu:~/environment/superails (main) $ rails db:drop
Dropped database 'superails_development'
Dropped database 'superails_test'
ubuntu:~/environment/superails (main) $ rails db:create
```

하지만 사용하지 않을 데이터베이스가 현재 있으므로 rails db:drop을 실행하여 사용하지 않을 데이터베이스들은 삭제

```
ubuntu:~/environment/superails (main) $ rails db:drop
Dropped database 'superails_development'
Dropped database 'superails_test'
ubuntu:~/environment/superails (main) $ rails db:create
Created database 'superails_development'
Created database 'superails_test'
ubuntu:~/environment/superails (main) $ rails db:migrate
```



이렇게 함으로서 user테이블을 가진 데이터베이스가 형성됨

```
ruby - "ip-172-31-4-6"  Gemfile  development.rb  application.html.erb  schema.rb
9 # migrations use external dependencies or application code.
10 #
11 # It's strongly recommended that you check this file into your version control system.
12
13 ActiveRecord::Schema.define(version: 2021_06_13_112726) do
14
15   # These are extensions that must be enabled in order to support this dat
16   enable_extension "plpgsql"
17
18   create_table "users", force: :cascade do |t|
19     t.string "email", default: "", null: false
20     t.string "encrypted_password", default: "", null: false
21     t.string "reset_password_token"
22     t.datetime "reset_password_sent_at"
23     t.datetime "remember_created_at"
24     t.datetime "created_at", precision: 6, null: false
25     t.datetime "updated_at", precision: 6, null: false
26     t.index ["email"], name: "index_users_on_email", unique: true
27     t.index ["reset_password_token"], name: "index_users_on_reset_password_token", unique: true
28   end
29
30 end
```



```
bash - "ip-172-31-4-6"  Gemfile  development.rb
1 class ApplicationController < ActionController::Base
2   before_action :authenticate_user!
3 end
```

application_controller.rb에

모든 액션을 수행할려면 로그인을 선행해야하도록 제약 추가 , rails s 실행 후 방문해보면 로그인 페이지로 리다이렉트됨

You need to sign in or sign up before continuing.

[Home](#) [Privacy](#) [Terms](#)

Log in

Email

Password

Remember me

Log in

[Sign up](#)

[Forgot your password?](#)



sign up 진행후,
bootstrap의 nav-bar를 이용한 디자인 구현 및

특정 페이지에 대해서는 로그인 할 필요 없이 사용 가능하게끔
수정

Navbar with Devise and Bootstrap (4, 5)

Mon, Dec 21, 2020 · Yaroslav Shimarov

When installing gem devise for a User model, add these links to your application:

Basic navigation:

```
<% if current_user %>
  <%= link_to current_user.email, edit_user_registration_path %>
  <%= link_to "Log out", destroy_user_session_path, method: :delete %>
<% else %>
  <%= link_to "Log in", new_user_session_path %>
  <%= link_to "Register", new_user_registration_path %>
<% end %>
```

bootstrap 4 Navbar:

```
<nav class="navbar navbar-expand-lg navbar-light bg-light">
  <a class="navbar-brand" href="/">
    <img alt="fas fa-flag" />
  Brand
  </a>
  <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarSupportedContent">
    <span class="navbar-toggler-icon"></span>
  </button>
  <div class="collapse navbar-collapse" id="navbarSupportedContent">
    <ul class="navbar-nav mr-auto">
      <%= link_to root_path, class: "nav-link #{'active font-weight-bold' if current_user}" %>
```

```
1  <!DOCTYPE html>
2  <html>
3    <head>
4      <title>Superails</title>
5      <meta name="viewport" content="width=device-width,initial-scale=1">
6      <%= csrf_meta_tags %>
7      <%= csp_meta_tag %>
8
9      <%= stylesheet_link_tag 'application', media: 'all', 'data-turbolinks-track': 'reload' %>
10     <%= javascript_pack_tag 'application', 'data-turbolinks-track': 'reload' %>
11     <%= stylesheet_pack_tag 'application', 'data-turbolinks-track': 'reload' %>
12   </head>
13
14   <body class="container">
15     <p class="notice"><%= notice %></p>
16     <p class="alert"><%= alert %></p>
17
18     <% if current_user %>
19       <%= link_to current_user.email, edit_user_registration_path %>
20       <%= link_to "Log out", destroy_user_session_path, method: :delete %>
21     <% else %>
22       <%= link_to "Log in", new_user_session_path %>
23       <%= link_to "Register", new_user_registration_path %>
24     <% end %>
25
26       <%= link_to 'Home', root_path %>
27       <%= link_to 'Privacy', privacy_path %>
28       <%= link_to 'Terms', terms_path %>
29     <% yield %>
30   </body>
31 </html>
```



You need to sign in or sign up before continuing.

[Log in](#) [Register](#) [Home](#) [Privacy](#) [Terms](#)

Log in

Email

Password

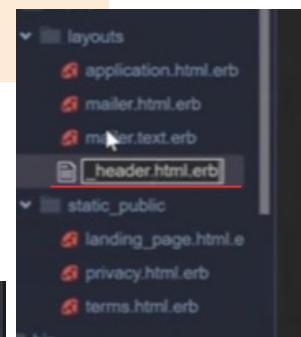
Remember me

Log in

[Sign up](#)

[Forgot your password?](#)

코드 가독성을 위해 partial로 nav부분을 분리



```
13
14  <body class="container">
15    <p class="notice"><%= notice %></p>
16    <p class="alert"><%= alert %></p>
17    <% render 'layouts/header' %>
18    <%= link_to 'Home', root_path %>
19    <%= link_to 'Privacy', privacy_path %>
20    <%= link_to 'Terms', terms_path %>
21    <% if current_user %>
22      <%= link_to current_user.email, edit_user_registration_path %>
23      <%= link_to "Log out", destroy_user_session_path, method: :delete %>
24    <% else %>
25      <%= link_to "Log in", new_user_session_path %>
26      <%= link_to "Register", new_user_registration_path %>
27    <% end %>
28    <% yield %>
29  </body>
30 </html>
```

```
1 <%= link_to 'Home', root_path %>
2 <%= link_to 'Privacy', privacy_path %>
3 <%= link_to 'Terms', terms_path %>
4 <% if current_user %>
5   <%= link_to current_user.email, edit_user_registration_path %>
6   <%= link_to "Log out", destroy_user_session_path, method: :delete %>
7 <% else %>
8   <%= link_to "Log in", new_user_session_path %>
9   <%= link_to "Register", new_user_registration_path %>
10 <% end %>
11
```



```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <title>SuperRails</title>
5     <meta name="viewport" content="width=device-width,initial-scale=1">
6     <%= csrf_meta_tags %>
7     <%= csp_meta_tag %>
8
9     <%= stylesheet_link_tag 'application', media: 'all', 'data-turbolinks-track': 'reload' %>
10    <%= javascript_pack_tag 'application', 'data-turbolinks-track': 'reload' %>
11    <%= stylesheet_pack_tag 'application', 'data-turbolinks-track': 'reload' %>
12   </head>
13
14   <body class="container">
15     <p class="notice"><%= notice %></p>
16     <p class="alert"><%= alert %></p>
17     <%= render 'layouts/header' %>
18     <%= yield %>
19   </body>
20 </html>
21
22
```



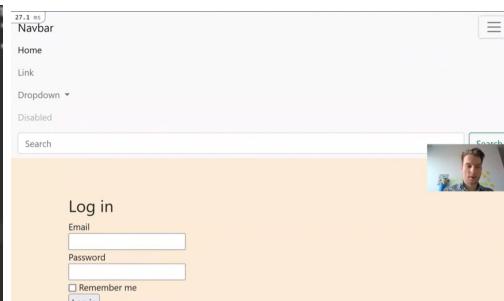
landing_page 액션에 대해서만 로그인하지 않고도 이용 가능하게 수정하기:

```
1 class StaticPublicController < ApplicationController
2   skip_before_action :authenticate_user!, only: [:landing_page]
3
4   def landing_page
5   end
6
7   def privacy
8   end
9
10  def terms
11  end
12 end
13
```

I

_header.html.erb 파일 bootstrap을 사용한 navbar 사용:

```
9   <%= stylesheet_link_tag 'application', >
10  <%= javascript_pack_tag 'application', >
11  <%= stylesheet_pack_tag 'application', >
12
13  </head>
14
15  <body>
16    <%= render 'layouts/header' %>
17    <%= render 'layouts/messages' %>
18    <div class="container">
19      <%= yield %>
20    </div>
21  </body>
22 </html>
23
```



uma - ip x Gemfile x develo x applic x applic x _head x _mess x static x +

```
<nav class="navbar navbar-expand-lg navbar-light bg-light">
<div class="container-fluid">
  <a class="navbar-brand" href="/">
    <i class="fas fa-flag"></i>
    Brand
  </a>
  <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarSupportedContent">
    <span class="navbar-toggler-icon"></span>
  </button>
  <div class="collapse navbar-collapse" id="navbarSupportedContent">
    <ul class="navbar-nav me-auto">
      <% link_to root_path, class: "nav-link #{'active fw-bold' if current_page?(root_path)}" do %>
        <div class="fa fa-home"></div>
      Home
      <% end %>
    </ul>
    <ul class="navbar-nav ms-auto">
      <% if current_user %>
        <li class="nav-item dropdown">
          <a class="nav-link dropdown-toggle" href="#" id="navbarDropdown" href="#">
            <div class="fa fa-user"></div>
            <b><%= current_user.email %></b>
          </a>
          <ul class="dropdown-menu" aria-labelledby="navbarDropdown">
            <% link_to edit_user_registration_path, class: "dropdown-item #{'active fw-bold' if %>
              <div class="fa fa-cog"></div>
              <b>Account settings</b>
            <% end %>
            <% link_to destroy_user_session_path, method: :delete, class: "dropdown-item" do %>
              <div class="fa fa-sign-out-alt"></div>
              <b>Sign out</b>
            <% end %>
          </ul>
        </li>
      <% end %>
    </ul>
  </div>
</div>
```

12:27 HTML (Ruby) Spaces: 2

27.1 ms

Brand Home Log in Sign up

StaticPublic#landing_page

Find me in app/views/static_public/landing_page.html.erb

439.9 ms

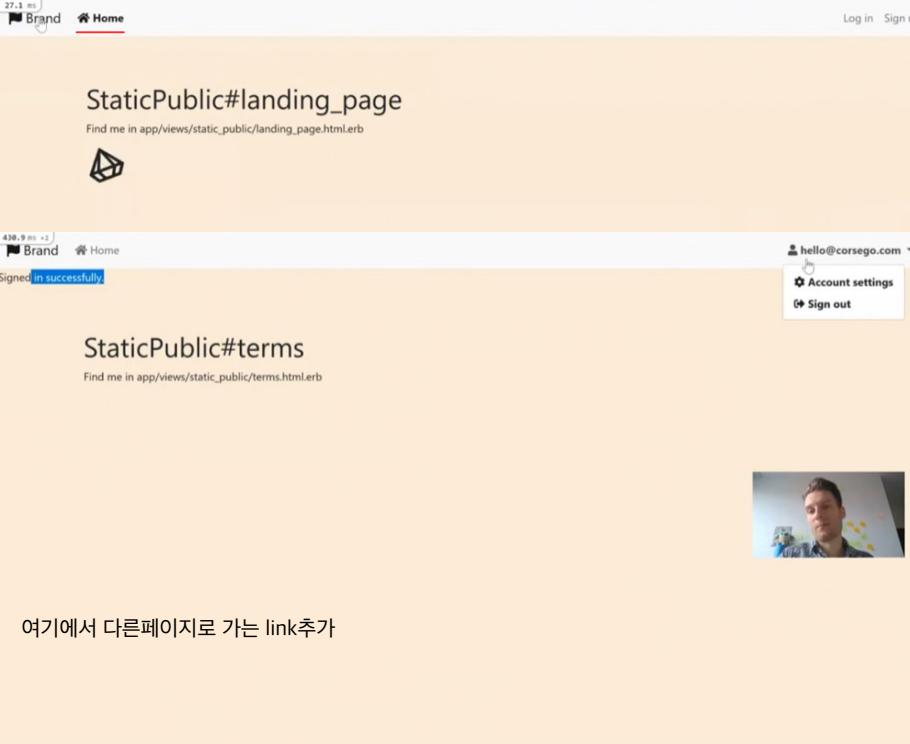
Brand Home

Signed in successfully.

StaticPublic#terms

Find me in app/views/static_public/terms.html.erb

hello@corsego.com Account settings Sign out



여기에서 다른페이지로 가는 link추가

```


Home


<% end %>
<%= link_to privacy_path, class: "nav-link #{'active fw-bold' if current_page?(privacy_path)}>
  <div class="fa fa-home"></div>
  Privacy
<% end %>
<%= link_to terms_path, class: "nav-link #{'active fw-bold' if current_page?(privacy_path)}>
  <div class="fa fa-home"></div>
  Privacy
<% end %>
```


ul class="navbar-nav ms-auto">

<% if current_user %>

- <li class="nav-item dropdown">
-
- <div class="fa fa-user"></div>
- <% current_user.email %>
-
- <ul class="dropdown-menu" aria-labelledby="navbarDropdown">
- <%= link_to edit_user_registration_path, class: "dropdown-item #{'active fw-bold' if current_page?(edit_user_registration_path)}>
- <div class="fa fa-cog"></div>
- Account settings
- <% end %>
- <%= link_to destroy_user_session_path, method: :delete, class: "dropdown-item" do %>
- <div class="fa fa-sign-out-alt"></div>
- Sign out
- <% end %>

<% else %>
<%= link_to "Log in", new_user_session_path, class: (5 Bytes) 20:21 HTML (Ruby) Spaces: 2 %>

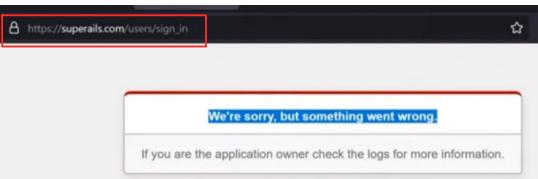
StaticPublic#privacy

Find me in app/views/static_public/privacy.html.erb

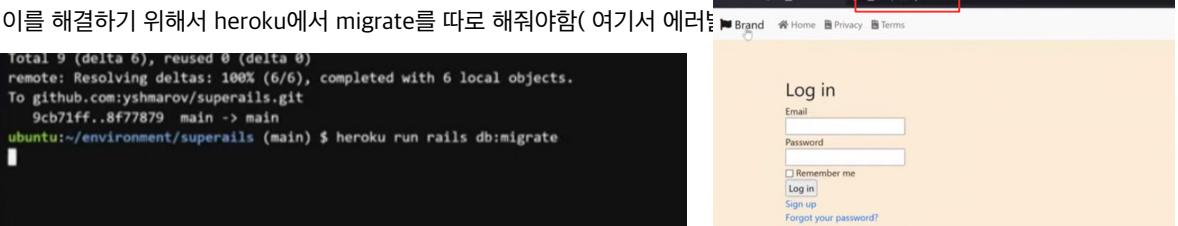
fontawsome에서 알맞는 아이콘 찾아서 사용하면됨



이상태로 production페이지를 방문해보면 다음과 같은 에러가 발생함



왜냐하면 로컬에서 user model을 새로 생성했기 때문에 production페이지에서는 migration이 반영되지 않았음



footer 추가하기(45:02) : 보통 <body> 태그 밑에 위치함

참조: <https://blog.corsego.com>

```

15  <body>
16    <%= render 'layout/header' %>
17    <%= render 'layout/footer' %>
18    <div class="container">
19      <%= yield %>
20    </div>
21  </body>
22  <footer></footer>
23</html>
```



heroku run rails db:migrate 명령어를 실행하고 나서도 production 환경에서 login기능이 동작하지 않는 이유는 Heroku가 SQLite 데이터베이스를 지원하지 않기 때문입니다. 대신 Heroku는 PostgreSQL 데이터베이스를 사용해야함.

참조 :https://negabaro.github.io/archive/config_database <https://m.blog.naver.com/ssang97/221848804509>

1. Gemfile에서 gem 'sqlite3'를 주석 처리하거나 삭제하고 gem 'pg'를 추가한 후 bundle install을 실행합니다.
2. psql을 사용하기 위해 PostgreSQL의 bin 디렉토리를 시스템의 PATH 환경 변수에 추가합니다. (이미 추가되어 있을 수도 있으니, 먼저 확인하세요.)
3. PostgreSQL의 사용자 이름(username)과 비밀번호(password)를 알고 있다면, 쉘에서 직접 접속할 수 있습니다. 만약 모른다면, PostgreSQL을 설치했을 때 설정했거나, 데이터베이스 관리자나 호스팅 서비스로부터 제공받았을 정보를 찾아야 합니다. 로컬에서 기본적으로 생성되는 사용자는 postgres입니다.

```
psql -U username -d databasename
```

4. 사용자 이름과 비밀번호를 알고 있다면, PostgreSQL 쉘에서 관리자 권한으로 로그인한 후, 사용자의 비밀번호를 변경할 수 있습니다.

```
ALTER USER username WITH PASSWORD 'newpassword';
```

5. config/database.yml 파일을 수정하여 PostgreSQL을 사용하도록 설정을 변경합니다. 사용자 이름과 비밀번호를 환경 변수를 통해 관리하는 것이 좋습니다.

6. 로컬 데이터베이스를 생성하고 마이그레이션을 실행합니다.

```
rails db:create  
rails db:migrate
```

7. 변경 사항을 테스트하고, 모든 것이 잘 작동하는지 확인합니다.
8. 마지막으로, 모든 설정이 완료되면 애플리케이션을 Heroku와 같은 클라우드 서비스에 배포할 수 있습니다. Heroku는 자동으로 DATABASE_URL을 설정해주므로 이를 config/database.yml에서 참조해야 합니다.

이 과정을 통해 Rails 애플리케이션을 SQLite에서 PostgreSQL로 전환할 수 있습니다.

```
default: &default  
adapter: postgresql  
encoding: unicode  
  
development:  
  <<: *default  
  database: 데이터베이스명  
  username: 사용자명  
  password: 패스워드  
  
  # Warning: The database defined as "test" will be erased and  
  # re-generated from your development database when you run "rake".  
  # Do not set this db to the same as development or production.  
test:  
  <<: *default  
  database: 데이터베이스명  
  username: 사용자명  
  password: 패스워드  
  
production:  
  <<: *default  
  database: 데이터베이스명  
  username: 사용자명  
  password: 패스워드
```

PostgreSQL의 사용자 이름과 비밀번호를 모르는 경우, 다음과 같은 방법으로 해결할 수 있습니다:

1. 기본 사용자 확인: PostgreSQL을 설치하면 기본적으로 postgres라는 사용자가 생성됩니다. 이 사용자를 사용하여 로그인을 시도해 볼 수 있습니다.
2. 비밀번호 재설정: 로컬에서 설치한 PostgreSQL에서 postgres 사용자의 비밀번호를 잊어버렸다면, 다음과 같은 단계로 비밀번호를 재설정할 수 있습니다:
 - PostgreSQL 서비스를 중지합니다.
 - PostgreSQL 설정 파일인 pg_hba.conf를 찾아서 엽니다. 이 파일은 일반적으로 PostgreSQL의 data 디렉토리 안에 있습니다.
 - pg_hba.conf 파일에서 모든 method를 trust로 설정하여 비밀번호 없이 접속할 수 있도록 변경합니다.
 - PostgreSQL 서비스를 다시 시작합니다.
 - 이제 psql을 사용하여 비밀번호 없이 postgres 사용자로 로그인합니다.
 - 로그인한 후, ALTER USER postgres WITH PASSWORD 'newpassword'; 명령어로 비밀번호를 재설정합니다.
 - pg_hba.conf 파일을 원래대로 되돌리고 (주로 md5나 scram-sha-256 설정), PostgreSQL 서비스를 다시 시작합니다.
3. 클라우드 서비스의 경우: 클라우드 서비스나 호스팅을 사용한다면, 서비스 제공업체의 관리 콘솔이나 지원을 통해 비밀번호를 재설정할 수 있습니다.
4. 환경 변수 확인: database.yml 파일이나 애플리케이션의 환경 설정에서 사용자 이름과 비밀번호가 환경 변수를 통해 설정되어 있는지 확인합니다.
5. 문서 확인: PostgreSQL을 설치할 때의 문서나 메모를 확인하여 사용자 이름과 비밀번호를 찾아봅니다.

위 단계를 따라도 접속 정보를 알아내거나 재설정하는 데 문제가 있다면, PostgreSQL 커뮤니티의 도움을 받거나 전문가의 조언을 구하는 것이 좋습니다. 데이터베이스 관리는 보안과 직결되므로 신중하게 처리해야 합니다.

config/database.yml 파일에서 사용자 이름과 비밀번호를 환경변수로 넣어야 보안상 안전함

환경변수 설정하는법

참조 : <https://velog.io/@rails/Rails-레일즈-데이터베이스-설정-방법-database.yml-작성>

- **adapter** : 사용할 DBMS 종류
- **pool** : 스레드 갯수
- **database** : 데이터베이스명(테이블들을 담고 있는 컨테이너)
- **timeout** : 쿼리 타임아웃 설정

yml 형식은 탭으로 데이터를 구분 하므로 indent에 주의 하자.

default를 상속 받아 development, test, production 세가지의 환경 설정을 꾸밀 수 있게 되어 있다.

아래 몇가지 옵션을 더 추가 할 수 있다.

- **host** : 데이터베이스 호스트 주소
- **port** : 데이터베이스 호스트 포트
- **encoding** : 사용할 문자 코드
- **username** : DBMS용 유저명
- **password** : DBMS용 비밀번호
- **socket** : 로컬에서 DB를 둘리는 경우 소켓 파일 주소

```
default: &default
  adapter: postgresql
  encoding: unicode
  pool: <%= ENV.fetch("RAILS_MAX_THREADS") { 5 } %>
  username: <%= ENV['POSTGRES_USER'] %>
  password: <%= ENV['POSTGRES_PASSWORD'] %>

development:
<:< *default
  database: myapp_development

test:
<:< *default
  database: myapp_test

production:
<:< *default
  database: myapp_production
  url: <%= ENV['DATABASE_URL'] %> # Heroku와 같은 서비스에서 제공하는 경우
```

2. 여기서 ENV['POSTGRES_USER']와 ENV['POSTGRES_PASSWORD']는 환경 변수의 이름입니다. 실제 환경변수 이름은 사용하는 환경에 맞게 지정해야 합니다.

3. .env 파일 사용:

로컬 개발에서는 dotenv 라이브러리를 사용하여 .env 파일에 환경변수를 저장하고, 이를 자동으로 로드할 수 있습니다. Gemfile에 dotenv-rails 점을 추가하고, .env 파일에 환경변수를 다음과 같이 추가합니다:

```
POSTGRES_USER=yourusername
POSTGRES_PASSWORD=yourpassword
```

3. 그리고 rails server를 시작할 때 dotenv가 이 파일을 읽어서 환경변수를 설정 합니다.

이 방법을 사용하면 database.yml 파일을 버전 제어 시스템에 커밋해도 민감한 정보가 노출되지 않습니다. 환경변수는 개발, 테스트, 프로덕션 환경 각각에 대해 별도로 설정할 수 있기 때문에 유연한 구성이 가능합니다.

Heroku의 Config Vars는 Heroku 애플리케이션에 대한 환경 변수를 설정하고 관리하는 기능입니다. 이들은 로컬 환경의 .env 파일에서 설정하는 환경 변수와 유사한 역할을 하며, 애플리케이션의 구성, 보안 정보, 외부 서비스의 API 키 등을 저장하는 데 사용됩니다.

Heroku Config Vars를 사용하는 방법은 다음과 같습니다:

1. Heroku 대시보드를 통해 설정하기:
 - Heroku 애플리케이션의 대시보드로 이동합니다.
 - “Settings” 탭을 클릭합니다.
 - “Config Vars” 섹션에서 “Reveal Config Vars”를 클릭합니다.
 - 여기서 새로운 키와 값을 추가하여 환경 변수를 설정할 수 있습니다.
2. Heroku CLI를 통해 설정하기:
 - Heroku CLI가 설치되어 있고, Heroku 계정으로 로그인된 상태라면, 커맨드 라인에서 다음과 같은 명령어를 사용하여 환경 변수를 설정할 수 있습니다:

```
heroku config:set KEY=VALUE -a app_name
```

Gemfile에 dotenv 점 추가:

Gemfile에 dotenv-rails 점을 개발과 테스트 그룹에 추가합니다. 프로덕션 환경에서는 보통 다른 방법(예: Heroku의 Config Vars)으로 환경 변수를 설정합니다.

3. 프로젝트 루트 디렉토리에 .env 파일을 생성합니다.

(Rails 프로젝트의 가장 상위 디렉토리를 의미합니다. 이는 프로젝트의 모든 파일과 폴더를 포함하는 디렉토리로, 예를 들어 Gemfile, app, config, db 등의 폴더와 파일이 위치)

이 파일에 환경 변수를 추가합니다.

2.
 - 여기서 KEY는 환경 변수의 이름, VALUE는 값, app_name은 Heroku 애플리케이션의 이름입니다.
3. 애플리케이션에서 Config Vars 사용하기:
 - Heroku에 설정한 환경 변수는 애플리케이션 코드 내에서 ENV['KEY']와 같은 형식으로 접근할 수 있습니다.
 - 예를 들어, DATABASE_URL 환경 변수는 Heroku Postgres를 사용하는 애플리케이션에서 데이터베이스 URL을 저장하는 데 사용됩니다.

Heroku Config Vars를 사용하면 중요한 정보를 안전하게 보관할 수 있으며, 애플리케이션 코드 내에 직접적으로 이러한 정보를 포함시키지 않아도 됩니다. 이는 보안 및 환경별 구성 관리 측면에서 매우 유용합니다.

1. 로컬 환경(개발 및 테스트):

- dotenv-rails 점을 사용하여 로컬 환경에서 .env 파일에 정의된 환경 변수를 로드합니다.
 - 이 점은 Rails 애플리케이션 시작 시 자동으로 .env 파일을 찾아서 환경 변수를 설정합니다.
 - .env 파일은 프로젝트의 루트 디렉토리에 위치하며, 보안상의 이유로 버전 관리 시스템에 포함되지 않아야 합니다.
2. 프로덕션 환경(예: Heroku):
 - Heroku에서는 dotenv-rails 점 대신 Config Vars를 사용하여 환경 변수를 관리합니다.
 - Heroku 대시보드에서 직접 설정하거나 Heroku CLI를 사용하여 Config Vars를 설정할 수 있습니다.
 - 이렇게 설정된 환경 변수는 Heroku에서 애플리케이션을 실행할 때 자동으로 사용할 수 있습니다.
 - Heroku 애플리케이션에서 ENV['KEY'] 형태로 환경 변수에 접근할 수 있습니다.

이러한 방식으로 로컬 환경과 프로덕션 환경에서 환경 변수를 분리하여 관리함으로써, 보안을 유지하고 환경별 구성은 쉽게 할 수 있습니다.

'dotenv-rails' 점을 로컬과 프로덕션 환경에서 모두 사용하지 않는 이유는 주로 관리, 보안, 그리고 환경 특성 때문입니다:

1. **환경 관리의 편의성**:

- 로컬 개발 환경에서는 '.env' 파일을 통해 환경 변수를 쉽게 관리하고 변경할 수 있습니다. 이는 개발자가 개별적으로 환경을 설정하고 빠르게 조정할 수 있는 유연성을 제공합니다.
- 반면, 프로덕션 환경에서는 환경 변수를 중앙화된 방식으로 관리하는 것이 더 효과적입니다. 클라우드 서비스 제공업체들은 이를 위해 자체적인 환경 변수 관리 시스템(예: Heroku의 Config Vars)을 제공합니다.

2. **보안 문제**:

- 로컬 환경에서 '.env' 파일을 사용하는 것은 편리하지만, 이 파일을 실수로 버전 관리 시스템에 포함시킬 위험이 있습니다. 이는 민감한 정보가 노출될 위험을 수반합니다.
- 프로덕션 환경에서는 이러한 실수를 방지하기 위해 환경 변수를 클라우드 서비스의 보안된 설정 영역에서 관리하는 것이 더 안전합니다.

3. **환경 특성의 차이**:

- 로컬 환경과 프로덕션 환경은 서로 다른 요구 사항과 설정이 필요합니다. 예를 들어, 데이터베이스 URL, API 키 등은 각 환경에 맞게 다르게 설정되어야 합니다.
- 클라우드 서비스에서는 환경 변수를 프로그래밍적으로 설정하고 관리할 수 있는 기능을 제공하여, 이러한 차이를 쉽게 관리할 수 있도록 합니다.

따라서, 로컬 환경에서는 'dotenv-rails' 점을 사용해 개발의 유연성을 높이고, 프로덕션 환경에서는 보안과 중앙화된 관리를 위해 클라우드 서비스의 환경 변수 관리 시스템을 사용하는 것이 일반적입니다.

```
production:
  <<: *default
  url: <%= ENV['DATABASE_URL'] %> # Heroku와 같은 서비스에서 제공하는 경우
```

Rails는 ENV['DATABASE_URL'] 환경 변수의 값을 사용하여 프로덕션

데이터베이스에 연결합니다. 이 변수는 Heroku에 의해 자동으로 설정되며, 애플리케이션의 데이터베이스 연결에 필요한 모든 정보를 포함합니다.

따라서, url 키를 통해 Heroku의 Config Var에서 DATABASE_URL 값을 직접 읽어서 사용하는 것이 맞습니다. 이 방식을 통해 코드 내에 사용자 이름이나 비밀번호와 같은 민감한 정보를 직접 입력하지 않아도 됩니다. Heroku는 이 정보를 안전하게 관리하고, 애플리케이션이 필요에 따라 접근할 수 있도록 합니다.

이때 'DATABASE_URL'은 add-on을 추가해야 Config Var에서 확인이 가능함.

.env 파일에서 환경 변수를 정의할 때 주변에 불필요한 공백이 있으면 안 됩니다. 올바른 형식은 KEY=VALUE 형태이며, 여기서 KEY는 환경 변수의 이름이고 VALUE는 해당 변수의 값을 의미합니다.

예를 들어, 올바른 환경 변수 설정은 다음과 같습니다:

```
POSTGRES_USER=postgres  
POSTGRES_PASSWORD=mypassword123
```

공백이 포함되면, 환경 변수의 값이 예상치 않게 " postgres"나 " mypassword123 "와 같이 공백이 앞뒤로 포함된 문자열로 로드될 수 있어, 이로 인해 애플리케이션이 데이터베이스에 연결하지 못하는 문제가 발생할 수 있습니다.

실제 환경 변수가 올바르게 로드되고 애플리케이션에서 사용될 수 있도록 dotenv-rails 점이 프로젝트에 추가되어야 하며,

개발환경에서 이를 로드하는 코드가 초기화 스크립트에 포함되어 있는지 확인해야 함.

config/application.rb 파일에서 Dotenv::Railtie.load 호출 되어야 함.

(그러나, dotenv-rails 점이 Gemfile에 포함되어 있고 bundle install이 실행되었다면, 이러한 수동 호출 없이도 환경 변수들은 자동으로 로드됩니다.)

추가적으로, 환경 변수가 실제로 의도한 값으로 설정되었는지 확인하기 위해 Rails 콘솔을 통해 환경 변수를 직접 확인해 볼 수 있습니다. 다음과 같이 Rails 콘솔을 실행하고 환경 변수를 조회할 수 있습니다:

```
rails console
```

그리고 콘솔에서 다음을 실행하여 환경 변수의 값을 확인합니다:

```
ENV['POSTGRES_USER']  
ENV['POSTGRES_PASSWORD']
```

```
C:\Users\com\Desktop\gunsuall\superails\some_project>rails c  
Loading development environment (Rails 7.0.8)  
irb(main):001:0> ENV['POSTGRES_USER']  
=> "postgres"  
irb(main):002:0> ENV['POSTGRES_PASSWORD']  
=> "mypassword123"  
irb(main):003:0>
```

이 값들이 .env 파일에 정의한 값과 일치하는지 확인해보세요. 만약 값이 다르다면, 환경 변수가 올바르게 설정되지 않은 것일 수 있습니다. 환경 변수가 올바르게 로드되지 않는다면 dotenv-rails 점이 제대로 설치되어 있고, 필요한 환경에서 로드되고 있는지 확인해야 합니다.

```
some_project > config > ! database.yml  
You, 18 minutes ago | 1 author (You)  
1 default: &default  
2   adapter: postgresql  
3   encoding: unicode  
4   # For details on connection pooling, see Rails configuration guide  
5   # https://guides.rubyonrails.org/configuring.html#database-pooling  
6   pool: <%= ENV.fetch("RAILS_MAX_THREADS") { 5 } %>  
7   username: <%= ENV['POSTGRES_USER'] %>  
8   password: <%= ENV['POSTGRES_PASSWORD'] %>  
9   #timeout: 5000  
10  
11 development:  
12   <<: *default  
13   database: postgresql  
14   username: <%= ENV['POSTGRES_USER'] %>  
15   password: <%= ENV['POSTGRES_PASSWORD'] %>  
16   # 추가 설정이 필요하다면 여기에 추가합니다.  
17   # 예를 들어, username, password, host 등을 명시할 수 있습니다  
18  
19 test:  
20   <<: *default  
21   database: postgresql  
22   username: <%= ENV['POSTGRES_USER'] %>  
23   password: <%= ENV['POSTGRES_PASSWORD'] %>  
24   # 추가 설정이 필요하다면 여기에 추가합니다.  
25  
26 production:  
27   <<: *default  
28   url: <%= ENV['DATABASE_URL'] %>
```

#	TYPE	DATABASE	USER	ADDRESS	METHOD
					pg_hba.conf` 설정
	local	all	all		md5
	host	all	all	127.0.0.1/32	md5
	host	all	all	::1/128	md5
	host	replication	all	127.0.0.1/32	md5
	host	replication	all	::1/128	md5

ActiveRecord::ConnectionNotEstablished

connection to server at "localhost" (::1), port 5432 failed: server closed the connection unexpectedly
This probably means the server terminated abnormally
before or while processing the request.

Extracted source (around line #87):

```
85      raise ActiveRecord::DatabaseConnectionError.hostname_error(conn_params[:hostname])
86    else
87      raise ActiveRecord::ConnectionNotEstablished, error.message
88    end
89  end
90
```

Rails.root: C:/Users/com/Desktop/gunsuall/superrails/some_project

C:\Users\com\Desktop\gunsuall\superrails\some_project>rake db:create
connection to server at "localhost" (::1), port 5432 failed: server clos
ed the connection unexpectedly

This probably means the server terminated abnormally
before or while processing the request.

Couldn't create 'postgresql' database. Please check your configuration.
rake aborted!

ActiveRecord::ConnectionNotEstablished: connection to server at "localho
st" (::1), port 5432 failed: server closed the connection unexpectedly (

ActiveRecord::ConnectionNotEstablished)

This probably means the server terminated abnormally
before or while processing the request.

C:/Users/com/.local/share/gem/ruby/3.2.0/gems/activerecord-7.0.8/lib/act
ive_record/connection_adapters/postgresql_adapter.rb:87:in `rescue in ne
w_client'

C:/Users/com/.local/share/gem/ruby/3.2.0/gems/activerecord-7.0.8/lib/act
ive_record/connection_adapters/postgresql_adapter.rb:77:in `new_client'

C:/Users/com/.local/share/gem/ruby/3.2.0/gems/activerecord-7.0.8/lib/act
ive_record/connection_adapters/postgresql_adapter.rb:37:in `postgresql_c
onnection'

C:/Users/com/.local/share/gem/ruby/3.2.0/gems/activerecord-7.0.8/lib/act
ive_record/connection_adapters/abstract/connection_pool.rb:656:in `publi

2 Col 1 (17 selected) Spaces: 4 UTF-8 CR/LF Properties ⌂ Go Live ⌂ Snell ⌂ Prettier ⌂

이미지에 표시된 오류 메시지는 "ActiveRecord::ConnectionNotEstablished"로, Rails 애플리케이션이 "localhost"의 5432 포트를 사용하는 PostgreSQL 서버에 연결을 시도했으나 서버가 연결을 예상치 못하게 닫았다는 것을 나타냅니다. 이는 서버가 요청을 처리하기 전이나 처리하는 동안 비정상적으로 종료되었을 가능성이 있음을 의미합니다.

이 문제를 해결하기 위해 다음 사항들을 확인해야 합니다:

1. **PostgreSQL 서비스 실행 여부**: PostgreSQL 서비스가 실제로 실행 중인지 확인하세요. 서비스가 실행되지 않았다면, 서비스를 시작해야 합니다. : 윈도우 검색 -> 시스템

2. **방화벽 설정**: PostgreSQL이 사용하는 포트(기본적으로 5432)가 Windows 방화벽에 의해 차단되지 않았는지 확인하세요.
: 방화벽 -> 고급보안이 포함된 windows Defender 방화벽 -> 인바운드 규칙에서 로컬포트 5432 확인

3. **`pg_hba.conf` 설정**: PostgreSQL의 인증 및 연결 설정 파일인 `pg_hba.conf`에 올바른 인증 방식이 설정되어 있는지 확인하세요. md5 또는 scram-sha-256 같은 비밀번호 기본 인증을 사용하도록 설정되어 있어야 합니다.

4. **데이터베이스 구성 파일**: `database.yml` 파일에서 올바른 데이터베이스 설정을 사용하고 있는지 확인하세요. 환경 변수가 정확하게 설정되어 있어야 합니다.

5. **환경 변수**: `.env` 파일에 설정된 환경 변수가 애플리케이션에서 올바르게 로드되고 있는지 확인하세요. `dotenv` 라이브러리를 사용하는 경우, 이 라이브러리가 Gemfile에 포함되어 있고 제대로 작동하고 있는지 검증하세요.

6. **PostgreSQL 로그**: PostgreSQL의 로그 파일을 확인하여 보다 구체적인 오류 메시지를 찾아보세요. 로그 파일에서는 연결 실패의 구체적인 원인에 대한 정보를 제공할 수 있습니다.

7. **PostgreSQL 서비스 재시작**: 설정 파일을 변경한 후에는 PostgreSQL 서비스를 재시작하여 변경 사항이 적용되도록 해야 합니다.

여기서 3, 4, 5를 위주로 살펴본 결과 원인을 찾을 수 있었음

먼저 명령프롬프트에서 psql을 바로 실행하기 위해 PATH 환경 변수에 PostgreSQL의 bin 디렉토리를 추가:

1. PostgreSQL bin 디렉토리 경로 찾기:

- PostgreSQL을 설치했을 때의 bin 디렉토리 경로를 찾습니다. 기본적으로 이 경로는 C:\Program Files\PostgreSQL\<version>\bin 형식이 됩니다. 여기서 <version>은 설치된 PostgreSQL의 버전을 나타냅니다.

2. 환경 변수 설정 창 열기:

- 검색 버튼이나 검색 창을 통해 '환경 변수' 또는 '시스템 속성'을 검색하거나, 제어판을 통해 '시스템'을 열고 '고급 시스템 설정'을 클릭합니다.

- '시스템 속성' 창에서 '환경 변수' 버튼을 클릭합니다.

3. PATH 환경 변수 편집:

- 시스템 변수 섹션에서 'Path' 또는 'PATH' 환경 변수를 찾고 '편집' 버튼을 클릭합니다.

- '새로 만들기' 버튼을 클릭하여 새 항목을 추가합니다.

- 복사한 PostgreSQL의 bin 디렉토리 경로를 붙여넣습니다.

- '확인'을 클릭하여 변경 사항을 저장합니다.

4. 시스템 재부팅 또는 새 명령 프롬프트 창 열기:

- 변경 사항을 적용하려면 시스템을 재부팅하거나 새로운 명령 프롬프트 창을 열어야 합니다.
- 재부팅 없이 바로 적용하려면, 모든 명령 프롬프트 창을 닫았다가 새로 열면 됩니다.

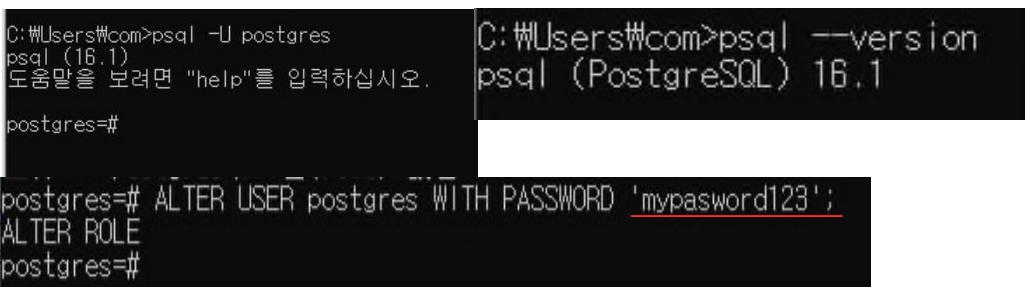
5. psql 명령어 테스트:

- 새로운 명령 프롬프트 창을 열고 psql --version을 입력하여 psql이 올바르게 설치되었는지 확인합니다.

path 환경변수 설정을 하고 나서 명령 프롬프트에서 다음 과 같은 명령어를 실행할 시,

```
psql -U <username> -d <database_name>
```

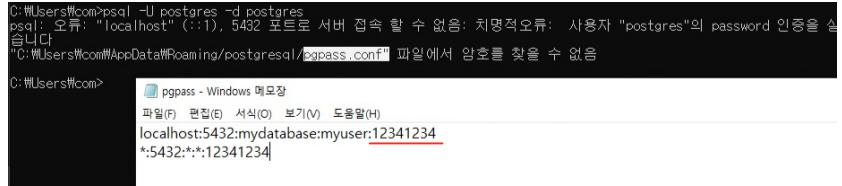
이 명령을 실행하면 시스템이 비밀번호를 입력하라는 프롬프트를 표시함. 여기에 해당 사용자 계정의 비밀번호를 입력하면 데이터베이스에 접근 가능. (종료 : \q 명령어 입력, 아직 비밀번호 설정 이전이어서 바로 접속됨)



```
C:\#Users\com>psql -U postgres
psql (16.1)
도움말을 보려면 "help"를 입력하십시오.

postgres=# ALTER USER postgres WITH PASSWORD 'mypassword123';
ALTER ROLE
postgres=#
C:\#Users\com>psql --version
psql (PostgreSQL) 16.1
```

이상태에서 명령프롬프트에서 "psql -U postgres -d postgres" 명령어를 실행 해보면 다음과 같은 오류발생:



```
C:\#Users\com>psql -U postgres -d postgres
오류: "localhost" (:1), 5432 포트로 서버 접속 할 수 없음: 치명적 오류: 사용자 "postgres"의 password 인증을 실패합니다
"C:\Users\com\AppData\Roaming/postgresql/pgpass.conf" 파일에서 암호를 찾을 수 없음

C:\#Users\com>
```



.pgpass 파일의 정보도 PostgreSQL 인터랙티브 터미널(psql)에서 설정했던 정보로 업데이트 해줘야함. .pgpass 파일은 PostgreSQL 클라이언트 애플리케이션이 비밀번호를 비대화적으로 읽을 수 있게하는데 사용됨. 즉 이 파일에 올바른 비밀번호가 있어야 클라이언트 애플리케이션을 통해 데이터베이스에 원활하게 인증하고 연결할 수 있음.

makefile

hostname:port:database:username:password

PostgreSQL의 psql 명령줄 도구에서 비밀번호를 입력할 때, 보안상의 이유로 화면에 비밀번호가 표시되지 않습니다. 즉, 비밀번호를 입력해도 화면상에 별표(*)나 다른 문자가 나타나지 않는 것이 정상

Gemfile

! database.yml

.env

X

some_project > .env

1 POSTGRES_USER=postgres

2 POSTGRES_PASSWORD=mypassword123

pgpass - Windows 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

localhost:5432:mydatabase:postgres:mypassword123

C:\Users\com>psql -U postgres -d postgres

psql 사용자의 암호:

psql (16.1)

도움말을 보려면 "help"를 입력하십시오.

postgres=# \list

이름	소유주	인코딩	로케일 제공자	Collate	데이터베이스 목록		ICU 로케일	ICU 룰	액세스 권한
					Ctype	Collate			
postgres	postgres	UTF8	libc	Korean_Korea.949	Korean_Korea.949				=c/postgres + postgres=CTc/postgres
template0	postgres	UTF8	libc	Korean_Korea.949	Korean_Korea.949				=c/postgres + postgres=CTc/postgres
template1	postgres	UTF8	libc	Korean_Korea.949	Korean_Korea.949				=c/postgres + postgres=CTc/postgres

PostgreSQL에서 현재 설정되어 있는 데이터베이스의 이름을 확인 하는 방법으로는

PostgreSQL 명령줄 도구 (psql)에서 '\' 또는 '\list' 명령어를 사용하여 모든 데이터베이스의 리스트를 볼 수 있음

현재 데이터베이스 리스트에서 postgres, template0, template1 세개의 데이터베이스가 있는데
template0, template1은 PostgreSQL이 내부적으로 사용하는 템플릿 데이터베이스임

따라서 현재 내가 사용하는 데이터베이스의 이름은 postgres이므로 .pgpass의 내용을 다음과 같이 수정해줘야함

pgpass - Windows 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

localhost:5432:postgres:postgres:mypassword123

이렇게 모든 설정을 한 후에 rails db:create , rails db:migrate를 실행하면 정상적으로 마이그레이션이 진행됨

C:\Users\com\Desktop\gunsuall\superails\some_project>rails db:create
Created database 'postgresql'
Database 'postgresql' already exists

C:\Users\com\Desktop\gunsuall\superails\some_project>rails db:migrate
== 20240102131518 DeviseCreateUsers: migrating =====
=====
-- create_table(:users)
--> 0.0113s
-- add_index(:users, :email, {:unique=>true})
--> 0.0032s
-- add_index(:users, :reset_password_token, {:unique=>true})
--> 0.0024s
== 20240102131518 DeviseCreateUsers: migrated (0.0180s) =====
=====

some_project > db > schema.rb
0 # GO SCHEMA LOAD . WHEN CREATING A NEW DATABASE. DDL/TABLES GO SCHEMA LOAD
1
2 # be faster and is potentially less error prone than running all of your
3 # migrations from scratch. Old migrations may fail to apply correctly if those
4 # migrations use external dependencies or application code.
5 #
6 # It's strongly recommended that you check this file into your version control
7 # system.
8 ActiveRecord::Schema[7.0].define(version: 2024_01_02_131518) do
9
10 # These are extensions that must be enabled in order to support this database
11 enable_extension "plpgsql"
12
13 create_table "users", force: :cascade do |t|
14 t.string "email", default: "", null: false
15 t.string "encrypted_password", default: "", null: false
16 t.string "reset_password_token"
17 t.datetime "reset_password_sent_at"
18 t.datetime "remember_created_at"
19 t.datetime "created_at", null: false
20 t.datetime "updated_at", null: false
21 t.index ["email"], name: "index_users_on_email", unique: true
22 t.index ["reset_password_token"], name: "index_users_on_reset_password_token"
23 end
24
25
26
27
28
29 end

C:\Users\com\Desktop\gunsuall\superails\some_project>[]

heroku run rails db:migrate 명령어를 다시 실행해보면

```
C:\Users\com\Desktop\gunsuall\superails\some_project>heroku run rails db  
:migrate  
Running rails db:migrate on somoproject... up, run.1801 (Basic)  
rails aborted!  
ActiveRecord::ConnectionNotEstablished: connection to server on socket "/var/run/postgresql/.s.PGSQL.5432" failed: No such file or directory ( ActiveRecord::ConnectionNotEstablished)  
      Is the server running locally and accepting connections on that socket?  
Caused by:  
PG::ConnectionBad: connection to server on socket "/var/run/postgresql/.s.PGSQL.5432" failed: No such file or directory (PG::ConnectionBad)  
      Is the server running locally and accepting connections on that socket?
```

와 같은 에러가 발생하는데 heroku에 add-on 설정을 하지 않아서 발생하는 문제임
Heroku에서는 Postgres 서비스를 별도로 설정해야함. Heroku Postgres 애드온이 프로젝트에 추가되었는지 확인

Heroku 웹 인터페이스를 통해 애드온을 추가하는 방법:

1. **Heroku 대시보드 접속**:

- 브라우저에서 [Heroku 대시보드](<https://dashboard.heroku.com>)에 접속합니다.
- Heroku 계정으로 로그인합니다.

2. **애플리케이션 선택**:

- 대시보드에서 애드온을 추가하고자 하는 애플리케이션을 클릭하여 선택합니다.

3. **리소스 탭으로 이동**:

- 애플리케이션 대시보드의 상단 메뉴에서 "Resources" 탭을 클릭합니다.

4. **애드온 추가**:

- "Resources" 탭에서 "Find more add-ons" 버튼을 클릭하거나 페이지 상단의 'Add-ons' 섹션에서 직접 'Add-on'을 검색할 수 있습니다.

- 필요한 애드온을 검색하고 선택합니다. 예를 들어, PostgreSQL을 추가하려면 "Heroku Postgres"를 검색할 수 있습니다.

5. **애드온 플랜 선택**:

- 애드온을 선택한 후에는 사용 가능한 플랜 목록이 표시됩니다.
- 원하는 플랜을 선택합니다. 예를 들어, 무료 플랜은 "Hobby Dev - Free"일 것입니다.
(현재는 없어짐)

6. **애드온 추가 확인**:

- 플랜을 선택한 다음, "Provision" 버튼을 클릭하여 애드온을 애플리케이션에 추가합니다.

7. **애드온 설정 확인**:

- 애드온이 성공적으로 추가되면, 자동으로 생성된 `DATABASE_URL` 와 같은 환경 변수들이 애플리케이션의 "Settings" 탭의 "Config Vars" 섹션에 표시됩니다.

위의 단계를 완료하면 Heroku 애플리케이션에 애드온이 추가되며, 필요한 환경 변수가 설정됩니다. 이제 애플리케이션은 이 환경 변수를 사용하여 애드온과 상호작용할 수 있습니다.

Overview Resources Deploy Metrics Activity Access Settings

Basic Dynos [Change Dyno Type](#)

web bin/rails server -p \${PORT:-5000} -e \$RAILS_ENV

 ~\$0.010/hour [Edit](#)

Add-ons

[Find more add-ons](#)The add-on heroku-postgresql has been installed. Check out the documentation in its [Dev Center](#) article to get started.

Quickly add add-ons from Elements

 Heroku Postgres [View details](#)Attached as DATABASE [Manage](#)Mini ~\$0.007/hour [Select](#)

Estimated Monthly Cost

\$12.00

Config Vars

Config vars change the way your app behaves.
In addition to creating your own, some add-ons come with their own.

Config Vars

[Hide Config Vars](#) DATABASE_URL

postgres://ctrjsibljelsmt:9d72cebbf063de

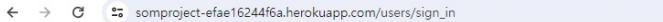
[Edit](#) [X](#) DEVISE_SECRET_KEY

ac3845c7a7d34f097f63b873f55189ca8c8fde908

[Edit](#) [X](#)

설정후 heroku run rails db:migrate 명령어 실행 후 production 환경에서 로그인 시도를 해보면 정상동작함:

```
C:\Users\com\Desktop\gunsuall\superails\some_project>heroku run rails db:migrate
Running rails db:migrate on somproject... up, run.6718 (Basic)
I, [2024-01-06T14:07:54.036198 #2] INFO -- : Migrating to DeviseCreateUsers (20240102131518)
== 20240102131518 DeviseCreateUsers: migrating =====
=====
-- create_table(:users)
-> 0.0078s
-- add_index(:users, :email, {:unique=>true})
-> 0.0030s
-- add_index(:users, :reset_password_token, {:unique=>true})
-> 0.0025s
== 20240102131518 DeviseCreateUsers: migrated (0.0136s) =====
=====
```





Log in

Email

Password

 Remember me[Log in](#)[Sign up](#)[Forgot your password?](#)[Sign in with GoogleOauth2](#)

코드 가독성을 위해서 _footer.html.erb 파일로 분리 및 bootstrap 스타일링 추가:

```
1 <div class="bg-light text-center">
2   <br>
3   THIS IS A FOOTER
4   <br>
5 </div>
6
```



```
<div class="bg-light text-center">
<br>
| <div class="container">
| <div class="row align-items-start">
|   <div class="col">
|     One of three columns
|   </div>
|   <div class="col">
|     <a href="#">Privacy</a>
|   <% end %>
|   <div class="col">
|     <a href="#">Terms</a>
|   <% end %>
| </div>
| </div>
| <br>
</div>
```



social account를 통한 로그인 기능 구현 (omniauth-google-oauth2 gem 사용 52:10~)

참조 : <https://github.com/zquestz/omniauth-google-oauth2>

README.md

For more details, read the Google docs: <https://developers.google.com/accounts/docs/OAuth2>

Installation

Add to your Gemfile :

```
gem 'omniauth-google-oauth2'
```

Then bundle install .

Google API Setup

- Go to <https://console.developers.google.com>
- Select your project.
- Go to Credentials, then select the "OAuth consent screen" tab on top, and provide an 'EMAIL ADDRESS' and a 'PRODUCT NAME'
- Wait 10 minutes for changes to take effect.

gem 'omniauth-google-oauth2' 을
gemfile에 추가한 후, bundle install 실행

config/initializers/devise.rb 에 주석처리 되어 있는 부분을

config.omniauth :google_oauth2, 'GOOGLE_CLIENT_ID', 'GOOGLE_CLIENT_SECRET', {}로 수정

```
# up on your models and hooks.
config.omniauth :google_oauth2, 'APP_ID', 'APP_SECRET'| You, now • Uncomm
```

The screenshot shows the Google Cloud API & Services dashboard under the 'API & 서비스' tab. A specific OAuth client ID is selected, showing its details. The client ID is named 'device_gem_test' and is described as using OAuth 2.0. It has a redirect URI of 'http://localhost:3000'. The 'Additional information' section shows the client ID was created on December 15, 2023, at 11:03:33 GMT+9.

승인된 클라이언트 원본

클라이언트 요청에 사용

URI 1: http://localhost:3000

+ URI 추가

승인된 리디렉션 URI

클라이언트 요청에 사용

URI 1: http://localhost:3000/users/auth/google_oauth2/callback

+ URI 추가

클라이언트 ID : 애플리케이션이 Google과 같은 서비스 제공자에게 자신을 식별하는 데 사용됩니다.

클라이언트 보안 비밀번호 : 인증을 요청할 때 사용하는 비밀 코드로, 이를 통해 애플리케이션이 서비스 제공자에게 실제로 등록된 신뢰할 수 있는 애플리케이션임을 증명합니다.

For use with requests from a browser

URIs *
https://superails.herokuapp.com
/899a2bca14d098c1453faac1fab6b.vfs.cloud9.eu-central-1.amazonaws.com
https://superails.com
https://www.superails.com

develop 와 production url을 입력

Authorized redirect URIs

For use with requests from a web server

URIs *
https://superails.herokuapp.com/users/auth/google_oauth2/callback
/vfs.cloud9.eu-central-1.amazonaws.com/users/auth/google_oauth2/callback
https://superails.com/users/auth/google_oauth2/callback
https://www.superails.com/users/auth/google_oauth2/callback

이때 client id와 client secret을 직접 입력할 시 , github등에 코드를 업로드하면 그대로 노출 되기 때문에 보안상 문제가 됨 따라서 EDITOR=vim rails credentials:edit 명령어 실행

```
# up on your models and hooks.
config.omniauth :google_oauth2, 'APP_ID', 'APP_SECRET'| You,
```

Visual studio의 기본 edtor는 메모장인데 수정사항이 정상적으로 반영되지 않는 문제점을 가지고 있음 때문에 vim을 설치한 후 환경변수의 시스템변수(S)에 Path에 를 vim으로 설정 (C:\Program Files\Vim\vim91) 하고 프로젝트의 터미널에 다음 명령어 실행:

```
set EDITOR="C:\Program Files\Vim\vim82\vim.exe"
rails credentials:edit 명령어를 실행
```

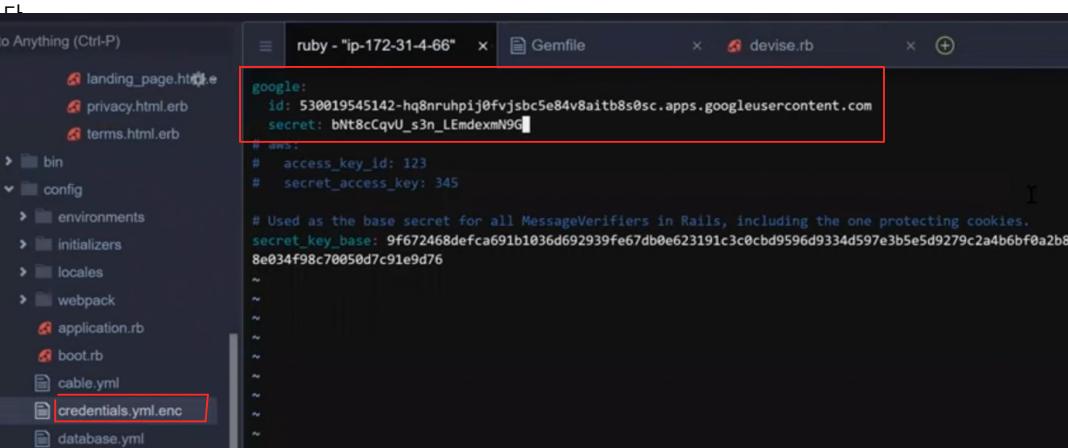
이 명령은 암호화된 credentials

Rails의 credentials:edit 태스크를 실행합니다.

Rails의 credentials 파일은 config/credentials.yml.enc에 저장되어 있으며, 암호화된 형태로 중요한 정보를 보관합니다. 예를 들어 API 키, 비밀번호, 시크릿 토큰 등이 이 파일에 포함될 수 있습니다. 이 파일은 config/master.key 또는 환경변수 RAILS_MASTER_KEY에 저장된 마스터 키로 암호화됩니다.

rails credentials:edit 명령을 실행하면 다음과 같은 과정이 일어납니다:

1. Rails는 config/credentials.yml.enc 파일의 암호화를 해제(master.key 사용)합니다.
2. 환경변수 EDITOR에 지정된 편집기가 열리고, 사용자는 해독된 내용을 편집할 수 있습니다.
3. 사용자가 편집을 마치고 편집기를 종료하면, Rails는 변경된 내용을 다시 암호화하여 config/credentials.yml.enc에 저장합니다.



실행한 후 i키를 눌러서 편집모드 진입후, 위와 같이 id ,secret를 입력한뒤 esc를 입력하여 편집모드 종료후 “ :wq ” 을 입력하여 편집기를 종료시킨다.

이제 제대로 저장되었는지 확인하기 위해 rails c 명령어를 실행

```
irb(main):004:0> Rails.application.credentials.dig(:google, :password)
=> "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"
irb(main):005:0> Rails.application.credentials.dig(:google, :id)
=> "837002718687-gd9lrdntjkf3nth15j1e2e6ue8n18kjn.apps.googleusercontent.com"
irb(main):006:0>
```

◀ ▶ ↻

config/initializers/devise.rb에 주석처리 다음과 같이 입력:

```
uby - "ip-172-31-4-66" x Gemfile x devise.rb • +  
format_forbidden, :format => :json  
HTTP method used to sign out a resource. Default is :delete.  
ut_via = :delete  
  
h  
omniauth provider. Check the wiki for more information on setting  
models and hooks.  
auth :github, 'APP_ID', 'APP_SECRET', scope: 'user,public_repo'  
|th :google_oauth2, Rails.application.credentials.dig(:google, :id), Rails.application.credentials.dig(:google, :secret)|
```

이제 Devise가 소셜 로그인 요청을 처리하기 위한 컨트롤러를 정의:

```
devise_project > config > routes.rb  
You, 1 second ago | 1 author (You)  
1 Rails.application.routes.draw do  
2   #devise_for :users  
3   devise_for :users, controllers: { omniauth_callbacks: 'users/omniauth_callbacks' }  
4   resources :users
```

controllers: { omniauth_callbacks: 'users/omniauth_callbacks' }

을 통해서 Omniauth를 통한 인증 과정(예:google, facebook 등을 통한 인증)이 성공적으로 완료된 후 실행될 액션을 사용자가 정의한 컨트롤러(users/omniauth_callbacks_controller.rb)에서 처리 할 수 있게 됩니다.

users폴더 와 그 내부에 omniauth_callbacks_controller.rb를 생성한 후에 devise gem깃허브 코드를 참고로 작성:

```
devise_project > app > controllers > users > omniauth_callbacks_controller.rb  
1 class Users::OmniauthCallbacksController < Devise::OmniauthCallbacksController  
2   def google_oauth2  
3     # You need to implement the method below in your model (e.g. app/models/user.rb)  
4     @user = User.from_omniauth(request.env['omniauth.auth'])  
5  
6     if @user.persisted?  
7       flash[:notice] = I18n.t 'devise.omniauth_callbacks.success', kind: 'Google'  
8       sign_in_and_redirect @user, event: :authentication  
9     else  
10      session['devise.google_data'] = request.env['omniauth.auth'].except('extra') # Removing extra as it can overflow some session stores  
11      redirect_to new_user_registration_url, alert: @user.errors.full_messages.join("\n")  
12    end  
13  end  
14 end
```

메소드 google_oauth2는 Google 로그인을 통해 반환된 데이터를 처리하는 역할을 합니다. 주요 내용은 다음과 같습니다:

1. `request.env['omniauth.auth']`를 통해 OmniAuth에서 제공하는 사용자 인증 정보를 가져옵니다.
2. `User.from_omniauth` 메소드를 호출하여 가져온 인증 정보를 기반으로 사용자 객체를 찾거나 생성합니다. (이 메소드는 보통 사용자 모델에 정의되어 있습니다.)
3. 사용자 객체(`@user`)가 영속적인지(즉, 데이터베이스에 저장되었는지) 확인합니다.
4. 만약 사용자 객체가 이미 데이터베이스에 저장되어 있다면(`if @user.persisted?`), 사용자를 로그인 상태로 만들고 알림 메시지와 함께 인증 성공 페이지로 리디렉션합니다.

event: :authentication는 Devise가 사용자가 성공적으로 인증된 후 실행하는 콜백 이벤트입니다. 이 이벤트는 sign_in_and_redirect 메소드에 전달되어 사용자가 로그인하면 리디렉션될 때 발생하는 이벤트를 나타냅니다.

sign_in_and_redirect @user 메소드는 Devise에서 제공하는 헬퍼 메소드로, 두 가지 주요 기능을 수행합니다:

1. sign_in - 주어진 @user 객체를 사용하여 사용자를 로그인 시킵니다. 즉, 사용자의 세션을 생성하고 유효한 사용자로 설정합니다.
2. redirect - 사용자를 지정된 경로나 기본적으로 설정된 경로로 리디렉션합니다. 이 경로는 보통 사용자가 로그인 후 보게 될 페이지입니다. 예를 들어, 사용자의 대시보드나 홈페이지가 될 수 있습니다.

5. 사용자 객체가 데이터베이스에 저장되지 않았다면, 인증 과정에서 얻은 정보를 세션에 저장(`session['devise.google_data'] = request.env['omniauth.auth'].except('extra')`)하고 사용자 생성에 메시지를 담아 회원가입 페이지로 리디렉션합니다.

이제 해당 모델(user)에 from_omniauth메소드를 정의:

```
user.rb M X routes.rb M omniauth_callbacks_controller.rb U
devise_project > app > models > user.rb
You, 1 second ago | 1 author (You)
1 class User < ApplicationRecord
2   # Include default devise modules. Others available are:
3   # :confirmable, :lockable, :timeoutable, :trackable and :omniauthable
4   devise :database_authenticatable, :registerable, :trackable,
5         :recoverable, :rememberable, :validatable, :omniauthable
6
7   def self.from_omniauth(access_token)
8     data = access_token.info
9     user = User.where(email: data['email']).first
10
11    unless user
12      user = User.create(name: data['name'],
13                           email: data['email'],
14                           password: Devise.friendly_token[0,20])
15    end
16  end
17  You, 1 second ago • Uncommitted changes
18  user
19
20 end
```

현재 users테이블에서 하나의 user를 특정 할 수 있는 요소가 email 뿐인상태이기 때문에 where에서 email만 조회

OAuth 제공자로부터 받은 데이터(access_token 인자의경우 `request.env['omniauth.auth']`)를 기반으로 사용자 객체를 찾거나 생성하는 기능을 합니다. 이 메소드는 OmniAuth의 auth 해시에서 받은 정보(예: 사용자 ID, 프로필 정보 등)를 사용하여 다음과 같은 작업을 수행합니다:

- 데이터베이스에서 제공된 인증 정보와 일치하는 사용자를 검색합니다.
- 일치하는 사용자가 이미 존재한다면 그 객체를 반환합니다.
- 일치하는 사용자가 없다면, 제공된 데이터로 새로운 사용자 객체를 생성하고, 필요에 따라 저장하기 전에 추가적인 검증이나 설정을 수행합니다.

즉, from_omniauth 메소드는 사용자가 OAuth를 통해 처음 로그인할 때 계정을 자동으로 생성하고자 할 때 유용합니다. 이 방식은 사용자가 별도의 회원가입 절차 없이 소셜로그인을 사용하여 애플리케이션에 바로 접근할 수 있게 해줍니다.

```
1 class User < ApplicationRecord
2   # Include default devise modules. Others available are:
3   # :confirmable, :lockable, :timeoutable, :trackable and :omniauthable
4   devise :database_authenticatable, :registerable,
5         :recoverable, :rememberable, :validatable, I
6         :omniauthable, :omniauth_providers: [:google_oauth2]
7
8   def self.from_omniauth(access_token)
9     data = access_token.info
10    user = User.where(email: data['email']).first
11
12    unless user
13      user = User.create(
14        email: data['email'],
15        password: Devise.friendly_token[0,20]
16      )
17    end
18    user
19  end
20
21
```

devise :omniauthable, :omniauth_providers: [:google_oauth2] 설정은 Devise를 사용하는 Rails 모델에 Omniauth를 통한 인증 기능을 추가하며, 사용할 OAuth 제공자를 명시적으로 지정합니다. 여기서 :google_oauth2는 Google을 OAuth 제공자로 사용하겠다는 것을 나타냅니다.

devise :omniauthable만 사용하는 경우, Omniauth 인증 기능이 활성화되지만 특정 제공자가 지정되지 않습니다. 따라서 앱에 여러 OAuth 제공자를 사용하려면, :omniauth_providers 옵션으로 각 제공자를 배열 형태로 추가해야 합니다.

devise :omniauthable, :omniauth_providers: [:google_oauth2] 설정을 함으로써, 사용자는 Google 계정으로 로그인할 수 있는 옵션이 추가되며, 이를 처리하기 위한 라우트와 컨트롤러 메소드를 구현할 수 있습니다.

ChatGPT

```
devise :omniauthable, omniauth_providers:  
[:google_oauth2] 설정을 한 후에, 사용자는 Google 계정을 사용해 인증할 수  
있도록 라우트와 컨트롤러 메소드를 구성해야 합니다.
```

라우트 설정 예시 (config/routes.rb):

```
Rails.application.routes.draw do  
  devise_for :users, controllers: { omniauth_callbacks: 'users/  
omniauth_callbacks' }  
  # 기타 라우트 설정...
```

위의 설정은 Devise에 의해 생성된 users/omniauth_callbacks 컨트롤러로 오는
OAuth 콜백을 라우팅합니다.

컨트롤러 메소드 예시 (app/controllers/users/
omniauth_callbacks_controller.rb):

```
class Users::OmniauthCallbacksController <  
Devise::OmniauthCallbacksController  
  def google_oauth2  
    # Omniauth에서 제공하는 정보를 기반으로 사용자를 찾거나 생성합니다.  
    @user = User.from_omniauth(request.env['omniauth.auth'])  
  
    if @user.persisted?  
      sign_in_and_redirect @user, event: :authentication  
      set_flash_message(:notice, :success, kind: 'Google') if  
is_navigational_format?  
    else  
      session['devise.google_data'] =  
request.env['omniauth.auth'].except('extra')  
      redirect_to new_user_registration_url, alert: 'Google 계정으로 로그인에 실  
폐하였습니다.'  
    end  
  end  
  
  # 기타 액션...
```



모델 메소드 예시 (app/models/user.rb):

```
class User < ApplicationRecord  
  devise :omniauthable, omniauth_providers: [:google_oauth2]  
  
  def self.from_omniauth(access_token)  
    data = access_token.info  
    user = User.where(email: data['email']).first  
  
    # 사용자가 존재하지 않으면 새로운 사용자를 생성합니다.  
    unless user  
      user = User.create(email: data['email'],  
                         password: Devise.friendly_token[0,20])  
    end  
    user  
  end  
end
```

from_omniauth 메소드는 OAuth 프로바이더로부터 전달받은 데이터를 사용하여
사용자를 찾거나 생성합니다. 사용자가 존재하지 않으면 임의의 비밀번호로 새로운
사용자를 데이터베이스에 추가합니다.

이 예시들은 Google OAuth2를 사용하여 Devise와 함께 사용자 인증을 처리하는 방법의
기본적인 구조를 보여줍니다.

evise_project > app > controllers > users > omniauth_callbacks_controller.rb

```
1 class Users::OmniauthCallbacksController < Devise::OmniauthCallbacksController  
2   def google_oauth2  
3     # You need to implement the method below in your model (e.g. app/models/user.rb)  
4     @user = User.from_omniauth(request.env['omniauth.auth'])  
5  
6     if @user.persisted?  
7       flash[:notice] = I18n.t 'devise.omniauth_callbacks.success', kind: 'Google'  
8       sign_in_and_redirect @user, event: :authentication  
9     else  
10       session['devise.google_data'] = request.env['omniauth.auth'].except('extra') # Removing extra as it can overflow some session  
11       redirect_to new_user_registration_url, alert: @user.errors.full_messages.join("\n")  
12     end  
13   end
```

sign_in_and_redirect @user, event: :authentication 메서드는 Ruby on Rails에서, 특히 인증을 위한 Devise 젬을 사용할 때, 인증
된 사용자를 로그인 시키고 지정된 경로로 리다이렉트하는 데 사용됩니다. @user 변수는 인증된 사용자 모델의 인스턴스를 나타냅니다.

event: :authentication 부분은 sign_in_and_redirect 메서드에 전달되는 옵션입니다. 이는 로그인을 “인증” 이벤트로 기록해야 함을 명
시합니다. 이것은 추적 목적이나 인증 이벤트에 대해 설정된 추가 콜백을 실행하기 위해 사용될 수 있습니다.

Could not authenticate you

회원가입

Log in

Email

Password

Remember me

[Log in](#)

[Sign up](#)

[Forgot your password?](#)

[Sign in with GoogleOauth2](#)

```
devise_project > app > views > devise > sessions > new.html.erb
1   <h2>Log in</h2>
2
3   <%= form_for(resource, as: resource_name, url: session_path(resource_name)) do |f| %>
4     <div class="field">
5       | <%= f.label :email %><br />
6       | <%= f.email_field :email, autofocus: true, autocomplete: "email" %>
7     </div>
8
9     <div class="field">
10    | <%= f.label :password %><br />
11    | <%= f.password_field :password, autocomplete: "current-password" %>
12   </div>
13
14   <% if devise_mapping.rememberable? %>
15     <div class="field">
16       | <%= f.check_box :remember_me %>
17       | <%= f.label :remember_me %>
18     </div>
19   <% end %>
20
21   <div class="actions">
22     | <%= f.submit "Log in" %>
23   </div>
24 <% end %>
25
26 <%= render "devise/shared/links" %>
```

render "devise/shared/links" 코드는 Devise에서 제공하는 뷰 템플릿 내에서 공통 링크를 렌더링하기 위해 사용됩니다. 이 코드가 있는 app/views/devise/shared/_links.html.erb 부분 템플릿에는 일반적으로 로그인, 회원가입, 비밀번호 재설정 등과 같은 공통 링크들이 포함되어 있습니다.

```
devise_project > app > views > devise > shared > _links.html.erb
```

```
1   <% if controller_name != 'sessions' %>
2     | <%= link_to "Log in", new_session_path(resource_name) %><br />
3   <% end %>
4
5   <% if devise_mapping.registerable? && controller_name != 'registrations' %>
6     | <%= link_to "Sign up", new_registration_path(resource_name) %><br />
7   <% end %>
8
9   <% if devise_mapping.recoverable? && controller_name != 'passwords' && controller_name != 'registrations' %>
10    | <%= link_to "Forgot your password?", new_password_path(resource_name) %><br />
11   <% end %>
12
13   <% if devise_mapping.confirmable? && controller_name != 'confirmations' %>
14     | <%= link_to "Didn't receive confirmation instructions?", new_confirmation_path(resource_name) %><br />
15   <% end %>
16
17   <% if devise_mapping.lockable? && resource_class.unlock_strategy_enabled?(:email) && controller_name != 'unlocks' %>
18     | <%= link_to "Didn't receive unlock instructions?", new_unlock_path(resource_name) %><br />
19   <% end %>
20
21   <% if devise_mapping.omniauthable? %>
22     | <% resource_class.omniauth_providers.each do |provider| %>
23       | <%= button_to "Sign in with #{OmniAuth::Utils.camelize(provider)}", omniauth_authorize_path(resource_name, provider), data: { turbo: false } %>
24     | <% end %>
25   <% end %>
```

저는 20대 후반의 남성으로 대한민국의 한 공기업에서 불로칼라 기술직으로 일하고 있습니다. 입사 4개월 차에 고장과 대출 간의 차별이 있으며, 오히려 고출력을 우대받는 현실과 대출자임에도 불구하고 더 낮은 연봉을 받는 것에 대한 회의감을 느낍니다. 이에 더 높은 고장과 넓은 시장에서 일할 수 있는 직종으로의 전환을 고민하게 되었습니다.

2023년 초에 정보처리기사 자격증을 공부해 6월에 취득했고, 이후 파이썬과 코딩 테스트 알고리즘을 학습했습니다. 하지만 이러한 공부 방식이 제가 원하는 병행과 일치하지 않음을 깨달았고, 그 때 저인의 스타트업에서 사용하는 '루비온레일즈'라는 웹 프레임워크에 대해 알게 되었습니다. 저인의 조언에 따라 2개월 동안 회사 일과 병행해가며 루비온레일즈로 페이징, 로그인, 터보스트림 등의 기능을 구현해보았고, 이제는 기사면 정도는 만들 수 있는 수준까지 능력을 키워왔습니다.

crud 게시판 정도는 만들 수 있는 수준까지 능력을 키웠습니다. 그러면 좀 유연히 유통브 알고리즘 (아마 제가 해외 유통브가 엄로드 한 룸비온레일즈 영상을 다수 시청하고 평소 일본 만화 및 애니메이션을 자주 봤던 것의 영향으로 추정) 현재 일본의 it문화가 높고, IT인력들의 공급이 부족해 블루오션이라는 영상을 보게 되었습니다. 평상시 저출산으로 인한 한국의 미래에 대해서 암담하게 생각하면서 저에게 뭔가 새로운 루트를 개척해낼 수 있다는 가뭄의 단비같은 영상이었습니다.

Could not authenticate you from GoogleOauth2 because "Authenticity error".

Log in

Email

Password

Remember me

[Sign up](#)
[Forgot your password?](#)
[Sign in with GoogleOauth2](#)

```
1  <h2>Log in</h2>
2
3  <%= form_for(resource, as: resource_name, url: session_path(resource_name)) do |f| %>
4      <div class="field">
5          <%= f.label :email %><br />
6          <%= f.email_field :email, autofocus: true, autocomplete: "email" %>
7      </div>
8
9      <div class="field">
10         <%= f.label :password %><br />
11         <%= f.password_field :password, autocomplete: "current-password" %>
12     </div>
13
14     <% if devise_mapping.rememberable? %>
15         <div class="field">
16             <%= f.check_box :remember_me %>
17             <%= f.label :remember_me %>
18         </div>
19     <% end %>
20
21     <div class="actions">
22         <%= f.submit "Log in" %>
23     </div>
24 <% end %>
25
26     <%= render "devise/shared/links" %>
```

render "devise/shared/links" 코드는 Devise에서 제공하는 뷰 템플릿 내에서 공통 링크를 렌더링하기 위해 사용됩니다. 이 코드가 있는

app/views/devise/shared/_links.html.erb 부분 템플릿에는 일반적으로 로그인, 회원가입, 비밀번호 재설정 등과

같은 공통 링크들이 포함되어 있습니다.

```
devise_project > app > views > devise > shared > _links.html.erb
1  <% if controller_name != 'sessions' %>
2      | <%= link_to "Log in", new_session_path(resource_name) %><br />
3  <% end %>
4
5  <% if devise_mapping.registerable? && controller_name != 'registrations' %>
6      | <%= link_to "Sign up", new_registration_path(resource_name) %><br />
7  <% end %>
8
9  <% if devise_mapping.recoverable? && controller_name != 'passwords' && controller_name != 'registrations' %>
10     | <%= link_to "Forgot your password?", new_password_path(resource_name) %><br />
11 <% end %>
12
13 <% if devise_mapping.confirmable? && controller_name != 'confirmations' %>
14     | <%= link_to "Didn't receive confirmation instructions?", new_confirmation_path(resource_name) %><br />
15 <% end %>
16
17 <% if devise_mapping.lockable? && resource_class.unlock_strategy.enabled?(:email) && controller_name != 'unlocks' %>
18     | <%= link_to "Didn't receive unlock instructions?", new_unlock_path(resource_name) %><br />
19 <% end %>
20
21 <% if devise_mapping.omniauthable? %>
22     | <%= resource_class.omniauth_providers.each do |provider| %>
23         | | <%= button_to "Sign in with #{OmniAuth::Utils.camelize(provider)}", omniauth_authorize_path(resource_name, provider), data: { turbo: false } %>
24     | <% end %>
25 <% end %>
```

만약 OmniAuth를 통해 구글 OAuth2를 설정하였다면, Devise는 해당 설정을 인식하고 자동으로

"Sign in with GoogleOAuth2" 같은 OAuth 인증 링크를 _links.html.erb 템플릿에 추가합니다. 이러한 링크들은

Devise와 OmniAuth 설정에 기반하여 자동으로 생성되며, 사용자가 해당 링크를 클릭할 때 OmniAuth를 통해 구글 로그인

절차가 시작됩니다. Devise가 이러한 링크를 어떻게 생성하는지는 devise/shared/_links.html.erb 파일을 확인하면 구체적인

로직을 볼 수 있습니다. 이 파일에는 다양한 조건문과 링크 생성 로직이 들어 있어 어떤 인증 옵션이 활성화되어

있는지에 따라 해당하는 링크를 렌더링합니다.

Could not authenticate you from GoogleOauth2 because "Authenticity error".

Log in

Email

Password

Remember me

[Sign up](#)

[Forgot your password?](#)

[Sign in with GoogleOauth](#)

```
gem "devise"
gem 'omniauth-google-oauth2'
gem "omniauth-rails_csrf_protection"
```

Google 계정으로 로그인

액세스 차단됨: 이 앱의 요청이 잘
못되었습니다

flavonoid37@gmail.com

이 앱에서 잘못된 요청을 전송했으므로 로그인할 수 없습니다. 나중에 다시 시도하거나 개발자에게 이 문제를 문의하세요. [이 오류에 관해 자세히 알아보기](#)

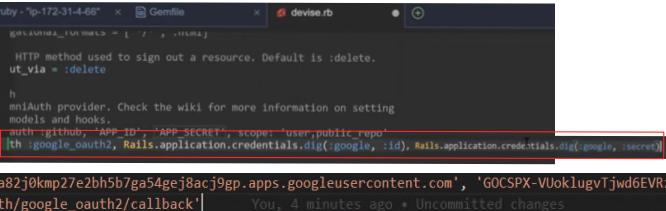
이 앱의 개발자인 경우 [오류 세부정보](#)를 참고하세요.

400 오류: redirect_uri_mismatch

```
config.omniauth :google_oauth2, '635217706972-da82j0kmp2e2bh5b7ga54gej8acj9gp.apps.googleusercontent.com', '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',
  redirect_uri: 'http://localhost:3000/users/auth/google_oauth2/callback'
```

'omniauth-rails_csrf_protection' 쟁을 사용하여 Rails 애플리케이션에 OmniAuth에 대한 CSRF 보호를 추가합니다.

로그인 시도중 다음과 같은 오류가 발생할 경우,
config/initializers/devise.rb 파일에 redirect_uri를
명시적으로 설정해야 할 수도 있습니다.



```
HTTP method used to sign out a resource. Default is :delete.
ut_via = :delete

h
h
  omniauth provider. Check the wiki for more information on setting
  models and hooks.
  auth :github, APP_ID, APP_SECRET, scope: 'user,public_repo'
  [th:google_oauth2, Rails.application.credentials.dig(:google, :id), Rails.application.credentials.dig(:google, :secret)]
```

위의 설정 해시에 redirect_uri를 추가할 수 있습니다.

localhost:3000/users/auth/google_oauth2/callback?state=16bcf86413e4317687161339ab21402747f76f1986332f&code=4%2FAf1chXmD_CZqITkkPqjRAzF4YntheZslazwi1aXbNmXS8JmDVYBRZAUiOe_GyIC.

NAVER | 번역이 완료 - Onlin... Gmail YouTube 카페 대학공과대학 - KO... (1) 수익 - Twitch 키프레리파이 No.1... 번들링이 라즈베리... 장령 데이터 다운로드... Binomial probabilit... #21 리스트

ActiveModel::UnknownAttributeError in Users::OmniauthCallbacksController#google_oauth2

unknown attribute 'name' for User.

```
Extracted source (around line #14):
12   # Uncount the section below if you want users to be created if they don't exist
13   unless user
14     user = User.create(name: data['name'],
15       email: data['email'],
16       password: Devise.friendly_token[0,20]
17   )
```

Rails.root: C:/Users/com/Desktop/gunsual/superails/some_project

Application Trace | Framework Trace | Full Trace

users/user (0.19ms) | tree (0.0ms)

users/controllers/users/omniauth_callbacks_controller.rb:4:in `google_oauth2'

User.create(name: data['name'])를 호출하는 코드 부분에서 발생했음, 즉 현재 데이터베이스에 name 컬럼이 존재하지 않아서 발생하는 문제임 따라서 rails generate migration AddNameToUsers name:string 명령어 실행후 rails db:migrate 명령어 실행하면 develop환경에서의 소셜로그인은 정상동작함 , 또는 name컬럼은 사용하지 않는다고 전제하고 models/user.rb를 다음과 같이 수정:

```
unless user
  user = User.create(name: data['name'],
    email: data['email'],
    password: Devise.friendly_token[0,20]
  )
end
```

```
unless user
  user = User.create(
    email: data['email'],
    password: Devise.friendly_token[0,20]
  )
end
```

한국어 ▾ 도움말 개인정보처리방침 악관

하지만 production 환경에서는 아직 동작안함, 몇가지 설정들을 추가해줘야함

한국어 ▾ 도움말 개인정보처리방침 악관

.gitignore 파일에 .env를 추가하면 .env 파일은 Git 버전 관리에 포함되지 않으므로 GitHub 같은 원격 저장소에 푸시되지 않습니다. Heroku 같은 클라우드 플랫폼은 Git 저장소에서 코드를 받아와 배포하기 때문에,.gitignore에 명시된 파일들은 Heroku에도 전송되지 않습니다.

그래서 Heroku에 배포할 때는 다음과 같은 방법으로 환경변수를 설정해야 합니다

```
C:\Users\com\Desktop\gunsual\superalails\some_project>heroku config:set client_id=REDACTED
Setting client_id and restarting ⚡ somproject... done, v23
client_id: 837002718687-gd9lrdntjfk3n3th15j1e2e6ue8n18kjn.apps.googleusercontent.com

C:\Users\com\Desktop\gunsual\superalails\some_project>heroku config:set secret_id=REDACTED
Setting secret_id and restarting ⚡ somproject... done, v24
secret_id: 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
```

Config Vars	Hide Config Vars
client_id	837002718687-gd9lrdntjfk3n3th15j1e2e6ue8n18kjn.apps.googleusercontent.com
client_secret	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
DATABASE_URL	postgres://ctrwsl1je1ejt:sd7tcbewfb03de
DEVISE_SECRET_KEY	ac3845c7fae334fb97f63b873f55189ca8cf8fe998
LANG	en_US.UTF-8
RACK_ENV	production
RAILS_ENV	production
RAILS_LOG_TO_STDOUT	enabled
RAILS_SERVE_STATIC_FILES	enabled
KEY	VALUE

로컬 환경과 프로덕션 환경에서 다른 설정을 사용하기 위해, Rails.env.production? 메소드를 사용하여 현재 환경이 프로덕션인지 여부를 확인할 수 있으며, 이를 통해 올바른 설정을 사용하도록 할 수 있습니다.

예를 들어, 다음과 같이 코드를 작성할 수 있습니다:

```
some_project > config > initializers > devise.rb
```

```
266 # config.navigational_formats = ['/*', :html, :turbo_stream]
267
268 # The default HTTP method used to sign out a resource. Default is :delete.
269 config.sign_out_via = :delete
270
271 # == OmniAuth
272 # # Add a new OmniAuth provider. Check the wiki for more information on setting
273 # up on your models and hooks.
274 # config.omniauth :github, 'APP_ID', 'APP_SECRET', scope: 'user,public_repo'
275 if Rails.env.production?
276   config.omniauth :google_oauth2, ENV['client_id'], ENV['client_secret'], {
277     redirect_uri: 'https://somoproject-efae16244f6a.herokuapp.com/users/auth/google_oauth2/callback'
278   }
279 else
280   config.omniauth :google_oauth2, Rails.application.credentials.dig(:google,:id), Rails.application.credentials.dig(:google,:password), {
281     redirect_uri: 'http://localhost:3000/users/auth/google_oauth2/callback'
282   }
283 end
```

```
irb(main):002:0> Rails.env.production?
=> false
irb(main):003:0> Rails.env.development?
=> true
irb(main):004:0>
```

Rails.env.production? 메소드는 Ruby on Rails 프레임워크에서 애플리케이션의 현재 실행 환경이 프로덕션(production)인지를 확인하는 방법을 제공합니다. Rails 애플리케이션은 일반적으로 세 가지 실행 환경을 가집니다: 개발(development), 테스트(test), 그리고 프로덕션(production).

이 메소드는 실행 중인 환경이 프로덕션 환경인 경우 true를 반환합니다. 즉, 애플리케이션이 실제 사용자에게 서비스되는 환경에서 동작 종일 때를 의미합니다. 다른 환경에서는 false를 반환합니다.

Rails.env.development?와 Rails.env.test?와 같은 메소드를 제공하여 개발 환경과 테스트 환경을 확인할 수 있도록 합니다. 이러한 메소드들은 주로 설정 파일이나 초기화 코드에서 환경에 따라 다른 설정값을 적용할 때 유용하게 사용됩니다.

설정 후 git push origin master 및 git push heroku master 명령어 실행후 heroku restart -a somoproject 명령어 실행 후 다시 시도하면:

We're sorry, but something went wrong.
If you are the application owner check the logs for more information.

```
계정 선택  
somoproject-efae16244f6a.herokuapp.com으로 이동  
  
프로젝트  
flavoron03@gmail.com  
ssdfl  
ssdfl1954@gmail.com
```

```
2024-01-07T05:48:40.576895+00:00 app[web.1]: [eb472493-9078-43f2-ac40-db6f22f2ad55] app/controllers/users/omniauth_callbacks_controller.rb:4:in `google_oauth2'  
2024-01-07T05:48:40.577944+00:00 heroku[router]: at=info method=GET path="/users/auth/google_oauth2/callback" host=somoproject-efae16244f6a.herokuapp.com request_id=eb472493-9078-43f2-ac40-db6f22f2ad55 fwd="118.216.251.49" dyno=web.1 connect=0ms service=201ms status=500 bytes=1827 protocol=https
```

```
some_project > app > controllers > users > omniauth_callbacks_controller.rb
```

```
You, 31 minutes ago | 1 author (You)
1 class Users::OmniauthCallbacksController < Devise::OmniauthCallbacksController
2   def google_oauth2
3     # You need to implement the method below in your model (e.g. app/models/user.rb)
4     @user = User.from_omniauth(request.env['omniauth.auth'])
5   end
You, 31 minutes ago
```

.from_omniauth는 user 모델에 정의된 메소드이므로 user모델에 메소드를 확인해서 수정

```

unless user
  user = User.create(name: data['name'],
    email: data['email'],
    password: Devise.friendly_token[0,20]
)
end

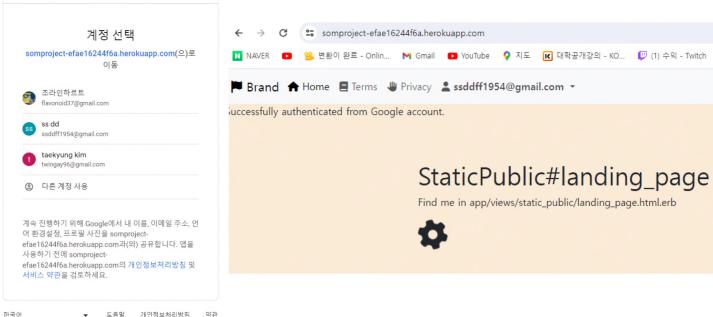
```

```

unless user
  user = User.create(
    email: data['email'],
    password: Devise.friendly_token[0,20]
)
end

```

설정 후 git push origin master 및 git push heroku master 명령어 실행후 heroku restart -a somproject 명령어 실행 후 다시 시도하면 정상동작함:



오류가 해결된 것과 관련하여, 가능성은 두 가지가 있습니다:

1. **name 칼럼 존재 여부**: 데이터베이스의 `users` 테이블에 `name` 칼럼이 없었을 경우, 그 칼럼에 데이터를 저장하려고 할 때 오류가 발생할 것입니다. 따라서 `name` 칼럼이 없다면, 해당 칼럼을 사용하는 코드(`data['name']`)를 제거하거나 주석 처리함으로써 오류를 해결할 수 있습니다.
2. **access_token.info 내 name 정보**: `access_token.info`에 `name`이라는 키에 해당하는 정보가 없을 경우, `data['name']`은 `nil`을 반환할 것이고, 이를 `User.create`로 전달하려고 할 때 오류가 발생할 수 있습니다. Google OAuth 응답에서 `name` 정보가 누락된 경우가 있을 수 있으며, 이로 인해 유효성 검사에 실패하거나, 예외가 발생할 수 있습니다.

위 두 경우 모두, `User.create` 호출 시 예외가 발생하지 않도록, 해당 필드가 데이터베이스에 존재하고, 전달받은 인증 데이터 내에 필요한 정보가 있는지를 확인해야 합니다.

해결 방법:

- `User` 모델에 `name` 필드가 있는지 데이터베이스 스키마를 확인하세요.
- `access_token.info` 내에 실제로 `name` 정보가 포함되어 있는지 확인하세요. 이 정보가 없다면, 빈 문자열이나 기본값을 사용하거나, 해당 필드를 사용하지 않도록 로직을 수정해야 합니다.

보통, 이러한 문제는 로컬 환경에서 개발하고 테스트할 때는 발견되지 않고, 실제 프로덕션 환경에 배포했을 때 나타나는 경우가 많습니다. 이는 프로덕션 환경의 데이터베이스 스키마와 로컬 환경의 스키마가 다를 수 있기 때문입니다. 또한 OAuth 제공자가 제공하는 데이터가 환경에 따라 다를 수도 있습니다.

SELECT * FROM users; postgresql-closed-96563

RESULTS UPDATED
2 rows in 147 ms January 7, 2024 16:14

[CSV](#) [JSON](#) [Table](#) [Chart](#)

	Id	Email	Encrypted_password	Reset_password_token	Reset_password_sent_at	Remember_created_at	Created_at	Updated_at
1	flavonoid37@gmail.com	\$2a\$12\$X2QfOmNjoU5VnL...	NULL	NULL	NULL	2024-01-07 06:42:07.696899	2024-01-07 06:42:07.696899	
2	ssddft1954@gmail.com	\$2a\$12\$5g71NMyH4RHvHU...	NULL	NULL	NULL	2024-01-07 06:47:42.874389	2024-01-07 06:47:42.874389	

Heroku의 웹 대시보드를 통해서도 데이터베이스를 조회할 수 있습니다. Heroku Dataclips를 사용하면 SQL 쿼리를 웹 인터페이스를 통해 실행하고 결과를 볼 수 있습니다. 다음은 웹 대시보드를 사용하여 데이터베이스를 조회하는 방법입니다:

1. 웹 브라우저에서 Heroku 대시보드로 이동하고 로그인합니다.
2. 대시보드에서 애플리케이션을 선택합니다.
3. “Resources” 탭을 클릭합니다.
4. “Add-ons” 섹션에서 “Heroku Postgres”를 찾고 클릭합니다.
5. Heroku Postgres 대시보드에서 “Dataclips”를 선택합니다.
6. “Create Dataclip”을 클릭하여 새 Dataclip을 생성하거나 기존 Dataclip을 실행합니다.
7. SQL 쿼리를 입력하고 “Run”을 클릭하여 결과를 확인합니다.

SELECT * FROM users; 쿼리문을 실행해보면 heroku데이터 베이스에 users 테이블에는 name 칼럼이 없다는 것을 확인 할 수 있음

1:01:25

Here's an example of an authentication hash available in the callback by accessing

```
request.env['omniauth.auth']:
```

```
{  
  "provider" => "google_oauth2",  
  "uid" => "100000000000000000000000",  
  "info" => {  
    "name" => "John Smith",  
    "email" => "john@example.com",  
    "first_name" => "John",  
    "last_name" => "Smith",  
    "image" => "https://lh4.googleusercontent.com/photo.jpg",  
    "urls" => {  
      "google" => "https://plus.google.com/+JohnSmith"  
    }  
  },  
  "credentials" => {  
    "token" => "TOKEN",  
    "refresh_token" => "REFRESH_TOKEN",  
    "expires_at" => 1496120719  
}
```

-> 이때 소셜로그인한 계정에 대한 정보 (access_token)은 다음과 같은 hash 구조를 가졌음

따라서 access_token내의 정보를 데이터베이스에 담기 위해서 현재 데이터베이스에는 존재하지 않는 컬럼드를 추가해줘야함 :

```
C:\Users\com\Desktop\gunsuall\superails\some_project>rails g migration add_omniauth_fields_to_users  
  invoke  active_record  
  create    db/migrate/20240107115358_add_omniauth_fields_to_users.rb
```

생성된 마이그레이션 파일에 다음과 같이 추가할 컬럼들을 작성 해주고 rails db:migrate 명령어 실행

```
some_project > db > migrate > 20240107115358 add_omniauth_fields_to_users.rb  
1  class AddOmniauthFieldsToUsers < ActiveRecord::Migration[7.0]  
2  def change  
3    | add_column :users, :provider, :string  
4    | add_column :users, :uid, :string  
5    | add_column :users, :name, :string  
6    | add_column :users, :image, :string  
7  end  
8 end  
9
```

```
C:\Users\com\Desktop\gunsuall\superails\some_project>rake db:migrate  
== 20240106182403 AddNameToUsers: migrating ======  
== 20240106182403 AddNameToUsers: migrated (0.000s) ======  
  
== 20240107115358 AddOmniauthFieldsToUsers: migrating ======  
-- add_column(:users, :provider, :string)  
-- 0.0020s  
-- add_column(:users, :uid, :string)  
-- 0.0007s  
-- add_column(:users, :name, :string)  
-- 0.0010s  
-- add_column(:users, :image, :string)  
-- 0.0006s  
== 20240107115358 AddOmniauthFieldsToUsers: migrated (0.0055s) =====
```

rails db:migrate 실행 후 user 모델에 정의 된 메소드(매 로그인마다 수행할 로직)에 추가된 컬럼들에대한 조회 및 생성 기능 추가

```
project > app > models > user.rb  
You, 1 minute ago | 1 author (You)  
class User < ApplicationRecord  
  # Include default devise modules. Others available are:  
  # :confirmable, :lockable, :timeoutable, :trackable and :omniauthable  
  devise :database_authenticatable, :registerable,  
         :recoverable, :rememberable, :validatable,  
         :omniauthable, :omniauth_providers => [:google_oauth2]  
  
  def self.from_omniauth(access_token)  
    user = User.where(email: access_token.info.email).first  
    # Uncomment the section below if you want users to be created  
    unless user  
      user = User.create(  
        email: access_token.info.email,  
        provider: access_token.provider,  
        name: access_token.info.name,  
        image: access_token.info.image,  
        password: Devise.friendly_token[0, 20]  
      )  
    end  
    user  
  end
```

Ruby에서 user를 메소드의 마지막 부분에 넣으려니 적는 것은 해당 메소드에서 마지막으로 평가된 표현식의 결과를 반환하려는 의도입니다. Ruby에서 메소드는 기본적으로 마지막에 평가된 표현식의 결과를 반환합니다. 이 경우 user 객체를 업데이트하거나 생성한 후 해당 user 객체를 반환하여 메소드를 호출한 다른 코드에서 사용할 수 있도록 합니다.

```

irb(main):001:0> User.last
User Load (1.0ms)  SELECT "users".* FROM "users" ORDER BY "users"."id" DESC LIMIT $1  [[{"LIMI
T": 1}]
=> #<User id: 3, email: "twingay96@gmail.com", created_at: "2024-01-07 06:36:45.486669000 +0000
", updated_at: "2024-01-07 06:36:45.486669000 +0000", provider: nil, uid: nil, name: nil, image
: nil>
irb(main):002:0>

```

rails c에서 조회시

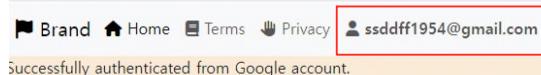
현재 user 테이블내 모든 데이터는 추가된 컬럼들에 대해서 nil 이므로 회원탈퇴를 하여서 데이터를 삭제해주고 다시 google로그인 해주면 추가한 컬럼들(name, provider, image)에 데이터가 생성된 것을 확인 할 수 있음

```

irb(main):001:0> User.all
User Load (0.8ms)  SELECT "users".* FROM "users"
=>
[#<User id: 3, email: "twingay96@gmail.com", created_at: "2024-01-07 06:36:45.486669000 +0000",
 updated_at: "2024-01-07 06:36:45.486669000 +0000", provider: nil, uid: nil, name: nil, image:
 nil>,
 #<User id: 6, email: "flavonoid37@gmail.com", created_at: "2024-01-07 12:21:20.551895000 +0000
", updated_at: "2024-01-07 12:21:20.551895000 +0000", provider: "google_oauth2", uid: nil, name
: "초라인하르트", image: "https://lh3.googleusercontent.com/a/ACg8ocJqS3BFLK..."]]
irb(main):002:0>

```

git add . -> git commit m "message" -> git push -u origin master -> git push heroku master 후
로컬에서 db의 스키마 수정되었으니 이를 heroku에도 적용 시키기 (heroku run rails db:migrate)
이후에 heroku restart -a somproject

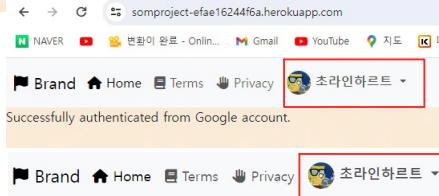


awsometag 대신에 google_oauth2 의 이미지 출력되게 변경

```

some_project > app > models > user.rb
10
11
12  unless user
13    user = User.create(
14      email: access_token.info.email,
15      provider: access_token.provider,
16      name: access_token.info.name,
17      image: access_token.info.image,
18      password: Devise.friendly_token[0,20]
19    )
20  end
21  user
22
23  def username
24    if name?
25      name
26    else
27      email.split('@').first
28    end
29  end
30

```



StaticPublic#landing_page

Find me in app/views/static_public/landing_page.html.erb



```

some_project > app > views > layouts > _header.html.erb
19
20
21  <%= link_to privacy_path, class: "nav-link #{'active font-weight-bold' if current_page?(privacy_path)}" do %>
22    <div class="fa-solid fa-hand"></div>
23    Privacy
24  <% end %>
25  </div>
26  <div class="navbar-nav ml-right">
27    <% if current_user %>
28      <li class="nav-item dropdown" id = "user_dropdown">
29        <a class="nav-link dropdown-toggle" href="#" id="navbarDropdown" type="button" data-bs-toggle="dropdown" a
30          <% if current_user.image? %>
31            <%= image_tag current_user.image, size: "30x30", class: "rounded-circle" %>
32          <% else %>
33            <div class="fa fa-user"></div>
34          <% end %>
35          <b><%= current_user.username %></b>
36        </a>
37      </li>
38    <% end %>
39  </div>
40

```

github로 로그인 기능 구현

참고 : <https://github.com/omniauth/omniauth-github>

The screenshot shows a browser window with tabs for 'superails - AWS Cloud9', 'superails', 'superails - GitHub | Heroku', and 'superails'. The main content is the GitHub Applications page for the 'superails' app, showing sections for 'Installation', 'Basic Usage', and 'Basic Usage Rails'. Below these are code snippets for OmniAuth::Builder and Rails.application.config.middleware.use. To the right, a terminal window titled 'bash - "ip-172-31-4-66"' shows the command 'bundle install' being run.

```
bundle install 실행
```

```
superails.com - AWS Cloud9 x Superails x superails - GitHub | Heroku x Superails x https://github.com/omniauth/omniauth-github x Superails
README.md
Application ID and Secret on the GitHub Applications Page.

Installation

Gemfile
43  # This file is copied to tmp/gems by keeping your application running
44  gem 'spring'
45 end
46
47 group :test do
48   # Adds support for Capybara system testing and selenium driver
49   gem 'capybara', '>= 3.26'
50   gem 'selenium-webdriver'
51   # Easy installation and use of web drivers to run system tests with browsers
52   gem 'webdrivers'
53 end
54
55 # Windows does not include zoneinfo files, so bundle the tzinfo-data gem
56 gem 'tzinfo-data', platforms: [:mingw, :mswin, :x64_mingw, :jruby]
57
58 gem 'devise'
59 gem 'omniauth-google-oauth2'
60 gem "omniauth-rails_csrf_protection"
61 gem 'omniauth-github', github: 'omniauth/omniauth-github', branch: 'master'
62
```

```
use OmniAuth::Builder do
  provider :github, ENV['GITHUB_KEY'], ENV['GITHUB_SECRET']
end
```

```
Rails.application.config.middleware.use OmniAuth::Builder do
  provider :github, ENV['GITHUB_KEY'], ENV['GITHUB_SECRET']
end
```

Basic Usage Rails

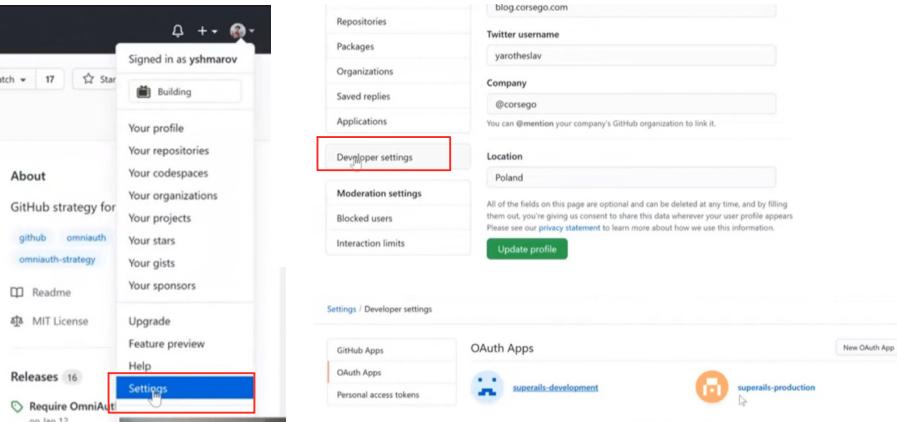
```
In config/initializers/github.rb
```

```
Rails.application.config.middleware.use OmniAuth::Builder do
  provider :github, ENV['GITHUB_KEY'], ENV['GITHUB_SECRET']
end
```

Github Enterprise Usage

```
provider :github, ENV['GITHUB_KEY'], ENV['GITHUB_SECRET'],
{
  client_options => {
    site => 'https://github.YOURDOMAIN.com/api/v3',
    authorize_url => 'https://github.YOURDOMAIN.com/login/oauth/authorize',
    token_url => 'https://github.YOURDOMAIN.com/login/oauth/access_token',
  }
}
```

git hub key secret 입력:



The screenshot shows a terminal window with the command 'EDITOR=vim rails credentials:edit' being run. The output shows the Vim editor with the command ':set paste' entered.

```
:set paste 입력
```

```
ubuntu:/environment/superails (main) $ EDITOR=vim rails credentials:edit
```

'`:set paste`' 명령은 Vim 편집기에서 사용합니다. 이 명령을 실행하면 'paste' 모드가 활성화되어, 외부에서 복사한 텍스트를 Vim에 붙어넣을 때 자동 들여쓰기나 기타 형식 관련 설정들이 적용되지 않도록 합니다. 이는 코드나 서식이 있는 텍스트를 붙여넣을 때 원본 형식을 유지하고자 할 때 유용합니다. 붙여넣기가 끝나면, '`:set nopaste`'를 사용하여 일반 모드로 돌아갈 수 있습니다.

:wq입력 해서 종료

```
ruby - "ip-172-31-4-66"  x  Gemfile  x  +  
google:  
  id: 530019545142-hq8nruhpij0fvjsbc5e84v8aitb8s0sc.apps.googleusercontent.com  
  secret: bNt8cCqvU_s3n_EMdexmN9G  
  
development:  
  github:  
    id: 64297b206d881ad2f93b  
    secret: ccb9a1f44dd857db960cb3e493f5801df3db8983  
production:  
  github:  
    id: cb033f68936978faaaef  
    secret: 32a29207e144c869eec66db4321884376c5a35b6  
  
# aws:  
#   access_key_id: 123  
#   secret_access_key: 345  
  
# Used as the base secret for all MessageVerifiers in Rails, including the one protecting cookies.  
secret_key_base: 9f672468defca691b1036d692939fe67db0e623191c3c0cbd9596d9334d597e3b5e5d9279c2a4b6bf0a2b  
49916d8daf5288e034f98c70050d7c91e9d76  
~
```

```
ubuntu:~/environment/superails (main) $ EDITOR=vim rails credentials:edit  
File encrypted and saved.  
ubuntu:~/environment/superails (main) $ rails c  
  running via Spring preloader in process 14195  
  loading development environment (Rails 6.1.3.2)  
.0.1 :001 > Rails.application.credentials.dig(:development, :github, :id)  
=> "64297b206d881ad2f93b"  
.0.1 :002 > Rails.application.credentials.dig(:production, :github, :id)  
=> "cb033f68936978faaaef"  
.0.1 :003 > Rails.application.credentials.dig(Rails.env.to_sym, :github, :id)  
=> "64297b206d881ad2f93b"  
.0.1 :004 > Rails.env.to_sym  
=> :development  
.0.1 :005 > |
```

```
273 |   # up on your models and hooks.  
274 |   config.omniauth :github, Rails.application.credentials.dig(Rails.env.to_sym, :github, :id), Rails.application.credentials.dig(Rails.env.to_sym, :google, :id), Rails.application.credentials.dig(:google, :id)  
275 |   config.omniauth :google_oauth2, Rails.application.credentials.dig(:google, :id), Rails.application.credentials.dig(:google, :id)  
276 |  
Application logo  
  
Drag & drop  
Upload new logo  
You can also drag and drop a picture from your computer.  
Application name *  
superails-production  
Something users will recognize and trust.  
Homepage URL *  
https://superails.com  
The full URL to your application homepage.  
Application description  
Application description is optional  
This is displayed to all users of your application.  
Authorization callback URL *  
https://superails.com/users/auth/github/callback  
Your application's callback URL. Read our OAuth documentation for more information.  
Update application  
  
class User < ApplicationRecord  
  # Include default devise modules. Others available are:  
  # :confirmable, :lockable, :timeoutable, :trackable and :omniauthable  
  devise :database_authenticatable, :registerable,  
         :recoverable, :rememberable, :validatable,  
         :omniauthable, omniauth_providers: [:google_oauth2, :github]  
  
  def self.from_omniauth(access_token)  
    user = User.where(email: access_token.info.email).first  
  
    unless user  
      user = User.create(  
        email: access_token.info.email,  
        password: Devise.friendly_token[0,20]  
      )  
    end  
  
    user.provider = access_token.provider  
    user.uid = access_token.uid  
    user.name = access_token.info.name  
    user.image = access_token.info.image  
    user.save  
  
    user  
  end  
  
  def username  
    if name?  
      name  
    else  
      email  
    end  
  end
```

```
uma - "ip-172-3 x Gemfile x devise.rb x user.rb x omniauth_ca +  
class Users::OmniauthCallbacksController < Devise::OmniauthCallbacksController  
  
| def google_oauth2  
|   # You need to implement the method below in your model (e.g. app/models/user.rb)  
|   @user = User.from_omniauth(request.env['omniauth.auth'])  
  
|   if @user.persisted?  
|     flash[:notice] = I18n.t 'devise.omniauth_callbacks.success', kind: 'Google'  
|     sign_in_and_redirect @user, event: :authentication  
|   else  
|     session['devise.google_data'] = request.env['omniauth.auth'].except('extra') # Remove extra info we don't care about  
|     redirect_to new_user_registration_url, alert: @user.errors.full_messages.join("\n")  
|   end  
| end  
end
```



```
puma - "ip-172-3 x Gemfile x devise.rb x user.rb x omniauth_ca +  
class Users::OmniauthCallbacksController < Devise::OmniauthCallbacksController  
  
1 | def google_oauth2  
2 |   handle_auth "Google"  
3 | end  
  
4 | def github  
5 |   handle_auth "Github"  
6 | end  
  
7 | def handle_auth(kind)  
8 |   # You need to implement the method below in your model (e.g. app/models/user.rb)  
9 |   @user = User.from_omniauth(request.env['omniauth.auth'])  
10 |  
11 |   if @user.persisted?  
12 |     flash[:notice] = I18n.t 'devise.omniauth_callbacks.success', kind: kind  
13 |     sign_in_and_redirect @user, event: :authentication  
14 |   else  
15 |     session['devise.auth_data'] = request.env['omniauth.auth'].except('extra') # Remove extra info we don't care about  
16 |     redirect_to new_user_registration_url, alert: @user.errors.full_messages.join("\n")  
17 |   end  
18 | end  
19 | end
```

Log in

Email

Password

Remember me

[Sign up](#)

[Forgot your password?](#)

[Sign in with GoogleOauth2](#)

[Sign in with GitHub](#)

```
node - "ip-172-3 x Gemfile x devise.rb x user.rb x omniauth_ca +  
ubuntu:~/environment/superails (main) $ heroku config:set RAILS_MASTER_KEY = 'cat config/master.key'  
> Warning: heroku update available from 7.52.0 to 7.54.1.  
* RAILS_MASTER_KEY is invalid. Must be in the format FOO=bar.  
ubuntu:~/environment/superails (main) $ heroku config:set RAILS_MASTER_KEY='cat config/master.key'  
> Warning: heroku update available from 7.52.0 to 7.54.1.  
Setting RAILS_MASTER_KEY and restarting ⚡ superails... done, v17  
RAILS_MASTER_KEY: cat config/master.key  
ubuntu:~/environment/superails (main) $ heroku run rails db:migrate  
heroku: command not found  
ubuntu:~/environment/superails (main) $ heroku run rails db:migrate  
> Warning: heroku update available from 7.52.0 to 7.54.1.  
Running rails db:migrate on ⚡ superails...  
|
```