


[개발환경] middleware 설정

☰ 구현 어려움 정도	☆☆
☑ 구현 완료	☑
📅 기간	@December 16, 2024
☑ 정리 완료	☑

Routing: Middleware

Learn how to use Middleware to run code before a request is completed.

 <https://nextjs.org/docs/pages/building-your-application/routing/middleware>



Routing: Middleware

미들웨어

- 목적
 - 인증된 사용자만 특정 경로에 접근할 수 있도록 제한
 - 인증되지 않은 사용자가 인증이 필요한 페이지에 접근할 경우 로그인 페이지로 리다이렉션
 - 인증된 사용자가 공개경로에 접근할 경우 홈페이지로 리다이렉션

작성 코드 설명

```
const { pathname } = request.nextUrl;  
const isPublicRoute = PUBLIC_ROUTES.includes(pathname);
```

- `pathname`: 현재 요청된 URL의 경로를 추출합니다.
- `isPublicRoute`: 요청된 경로가 공개경로 목록에 포함되는지 확인합니다. `PUBLIC_ROUTES`는 인증 없이 접근할 수 있는 페이지들입니다.

```
const cookie = (await cookies()).get('session')?.value;  
const session = await verify(cookie);
```

- `cookies()`: 현재 요청의 쿠키를 가져옵니다. `session` 쿠키를 찾고 그 값을 `cookie` 변수에 할당합니다.
- `verify(cookie)`: 쿠키의 세션 값을 검증합니다. `verify` 함수는 세션이 유효한지 확인하고 유효한 경우 세션 정보를 반환합니다.

```
if (!isPublicRoute && !session) {  
  return NextResponse.redirect(new URL(AUTH_ROUTES.LOGIN, request.nextUrl));  
}
```

- 비공개 경로에서 인증되지 않은 사용자의 접근 차단
- `!isPublicRoute && !session`: 요청된 경로가 비공개 경로이고 사용자가 인증되지 않은 경우가 참(true)일 경우 `NextResponse.redirect()` 를 사용해 로그인 페이지로 리다이렉션합니다.
- `AUTH_ROUTES.LOGIN`: 로그인한 페이지의 URL을 나타내는 상수입니다.

```
if (isPublicRoute && session) {  
  return NextResponse.redirect(new URL(BASE_URL, request.nextUrl));  
}
```

- 공개 경로에서 인증된 사용자의 접근 차단

- `isPublicRoute` && `session`: 요청된 경로가 공개 경로이고 사용자가 인증된 상태인 경우가 참일 경우, `NextResponse.redirect()` 를 사용해 홈페이지 (BASE_URL)로 리디렉션합니다.
- 인증된 사용자가 로그인된 상태로 공개 경로에 접근하는 것을 방지합니다.

```
return NextResponse.next();
```

- 인증이 필요한 경로에 인증된 사용자가 접근하거나, 공개 경로에 인증되지 않은 사용자가 접근하는 경우가 아니라면 요청을 계속 진행합니다.
- `NextResponse.next()` 는 미들웨어에서 요청을 차단하지 않고 요청이 정상적으로 진행되도록 합니다.

미들웨어 config 설정

```
export const config = {
  matcher: [
    '/(?!api|_next/static|_next/image|favicon.ico|sitemap.xml|robots.txt).*',
  ],
};
```

- `matcher`: 미들웨어가 적용될 URL 패턴을 설정합니다.
 - 여기서는 `api`, `_next/static`, `_next/image`, `favicon.ico`, `sitemap.xml`, `robots.txt` 를 제외한 모든 경로에 미들웨어를 적용하도록 설정되어 있습니다.
 - 예를들어, `api`로 시작하는 경로나 Next.js의 정적 파일 경로 (`_next/static`, `_next/image`) 등은 미들웨어에서 처리하지 않고 무시합니다.

장점

- 사용자 인증 관리
 - 인증된 사용자만 특정 페이지에 접근할 수 있도록 제한
- 자동 리디렉션
 - 인증되지 않은 사용자가 인증에 필요한 페이지에 접근할 경우 자동으로 로그인 페이지로 리디렉션할 수 있습니다.
- 세션관리
 - `session` 쿠키를 통해 사용자 인증을 처리하고 서버에서 세션을 검증해 인증 상태를 관리합니다.
- 비공개 경로 보호
 - 인증된 사용자만 비공개 경로에 접근할 수 있도록 하고 공개 경로에서는 인증된 사용자에게 리디렉션을 수행

주의할 점

- 세션 로직 검증 꼭!
 - `verify(cookie)` 함수에서 세션을 제대로 검증하지 않으면 인증된 사용자라도 비공개 경로에 접근할 수 없게 됩니다.
- 성능 이슈
 - 세션 검증이 요청마다 수행되므로 세션 검증 속도나 네트워크 성능에 따라 페이지 로딩이 느릴 수 있습니다.

전체 코드

```
// middleware.ts

import { NextResponse } from 'next/server'
import type { NextRequest } from 'next/server'
import { AUTH_ROUTES, BASE_URL, PUBLIC_ROUTES } from '@/constants/routes';
import { cookies } from 'next/headers';
import { verify } from '@/actions/sessions';
```

```

export async function middleware(request: NextRequest) {
  const { pathname } = request.nextUrl;
  const isPublicRoute = PUBLIC_ROUTES.includes(pathname);

  const cookie = (await cookies()).get('session')?.value;
  const session = await verify(cookie);

  if (!isPublicRoute && !session) {
    return NextResponse.redirect(new URL(AUTH_ROUTES.LOGIN, request.nextUrl));
  }

  if (isPublicRoute && session) {
    return NextResponse.redirect(new URL(BASE_URL, request.nextUrl));
  }

  return NextResponse.next();
}

export const config = {
  matcher: [
    /*
     * Match all request paths except for the ones starting with:
     * - api (API routes)
     * - _next/static (static files)
     * - _next/image (image optimization files)
     * - favicon.ico, sitemap.xml, robots.txt (metadata files)
     */
    '/((?!api|_next/static|_next/image|favicon.ico|sitemap.xml|robots.txt).*)',
  ],
}

```

```

// routes.ts
export const BASE_URL = "/";

export const AUTH_ROUTES = {
  LOGIN: '/login',
  SIGN_UP: '/signup',
}

export const PUBLIC_ROUTES = [AUTH_ROUTES.LOGIN, AUTH_ROUTES.SIGN_UP];

```