Brevity Backend API Documentation

Base URL: https://brevitybackend.onrender.com/api/

# Authentication Routes

## Register User - POST /auth/register

Headers:  
Content-Type: multipart/form-data

Form-data Fields:  
- displayName (String, required)  
- email (String, required)  
- password (String, required)  
- profileImage (File, optional)

Success Response Example:

{  
 "success": true,  
 "message": "User registered successfully",  
 "data": {  
 "user": { "\_id": "...", "displayName": "John Doe", "email": "john@example.com" },  
 "accessToken": "...",  
 "refreshToken": "..."  
 }  
}

## Login - POST /auth/login

Body:  
{  
 "email": "john@example.com",  
 "password": "supersecure"  
}

Success Response Example:

{  
 "success": true,  
 "message": "Login successful",  
 "data": {  
 "user": { "\_id": "...", "displayName": "...", "email": "..." },  
 "accessToken": "...",  
 "refreshToken": "..."  
 }  
}

## Logout - POST /auth/logout

Headers:  
Authorization: Bearer <accessToken>

Optional Body:  
{  
 "refreshToken": "..."   
}

Success Response Example:

{  
 "success": true,  
 "message": "Logout successful"  
}

## Get Current User - GET /auth/me

Headers:  
Authorization: Bearer <accessToken>

Response Example:

{

"success": true,

"data": {

"user": {

"\_id": "abc123",

"displayName": "John Doe",

"email": "john@example.com",

"profileImage": {

"url": "https://res.cloudinary.com/.../profile.jpg",

"publicId": "newsai/profiles/abc123"

}

}

} }

## Resend Email Verification - POST /auth/resend-verification

Headers:  
Content-Type: application/json

Body:

{

"email": "john@example.com"

}

Success Response Example:

{

"success": true,

"message": "Verification email sent successfully"

}

## Verify Email - GET /auth/verify-email

Query Parameters:

* token (String, required) — the verification token (controller decodes this token to obtain the email & verification token)

**Note:** On success this route **returns an HTML page** (rendered with helper/html.helper). On failure it returns JSON error.

Success (HTTP 200) — HTML page is returned (example not JSON):

<!-- HTML page content confirming verification (rendered server-side) -->

Failure Example (JSON):

{

"success": false,

"message": "Invalid or expired email verification token"

}

## Forgot Password - POST /auth/forgot-password

Headers:  
Content-Type: application/json

Body:

{

"email": "john@example.com"

}

Success Response Example:

{

"success": true,

"message": "Password reset link sent to your email"

}

## Reset Password - POST /auth/reset-password

{

"email": "john@example.com",

"token": "resetTokenHere",

"newPassword": "newStrongPassword123"

}

Behavior & Success Response Example:

* Controller validates user, validates token via user.comparePasswordResetToken(token), checks passwordResetExpires, then replaces password and clears reset fields.

{

"success": true,

"message": "Password has been reset successfully"

}

Failure examples from controller:

// missing fields

{

"success": false,

"message": "Email, token, and new password are required"

}

// invalid/expired token

{

"success": false,

"message": "Invalid or expired password reset token"

}

# Bookmark Routes

**Auth required:** Authorization: Bearer <accessToken>

## Get Bookmarks - GET /bookmarks

Headers:  
Authorization: Bearer <accessToken>

Behavior:

* If no bookmarks found → returns 404 with a JSON message.
* If bookmarks exist → returns a plain JSON **array** of articles (controller maps timeAgo onto each article).

No-bookmarks Example (HTTP 404):

{

"message": "No bookmarks found"

}

Bookmarks Found Example (HTTP 200) — plain array response:

[

{

"title": "Example Article",

"description": "Short desc...",

"url": "https://example.com/article",

"urlToImage": "https://example.com/image.jpg",

"publishedAt": "2025-08-10T12:00:00.000Z",

"sourceName": "Source",

"author": "Author Name",

"content": "...",

"timeAgo": "2 days ago"

},

...

]

## Toggle Bookmark - POST /bookmarks

Headers:  
Authorization: Bearer <accessToken>  
Content-Type: application/json

**Body fields** (controller reads these to create a bookmark if it doesn't exist; it uses url to check existence):

* title
* description
* url (used to check existing bookmark)
* urlToImage
* publishedAt
* sourceName
* author
* content

Responses from controller:

* If bookmark existed and was removed:

{

"message": "Bookmark removed"

}

* If bookmark was created (HTTP 201):

{

"message": "Bookmark added",

"article": {

"\_id": "...",

"title": "...",

"description": "...",

"url": "...",

"urlToImage": "...",

"publishedAt": "...",

"sourceName": "...",

"author": "...",

"content": "...",

"timeAgo": "..."

}

}

# User Routes

## Update Profile - PUT /users/profile

Headers:  
Authorization: Bearer <accessToken>  
Content-Type: multipart/form-data

Form-data Fields:  
- displayName (String, optional)  
- profileImage (File, optional)

Success Response Example:

{  
 "success": true,  
 "message": "Profile updated successfully",  
 "data": {  
 "user": { "\_id": "...", "displayName": "...", "email": "...", "profileImage": {...} }  
 }  
}

## Delete Profile Image - DELETE /users/profile/image

Headers:  
Authorization: Bearer <accessToken>

Success Response Example:

{  
 "success": true,  
 "message": "Profile image deleted successfully"  
}

# Health Check

## Health Check - GET /health

Response Example:

{  
 "success": true,  
 "status": "OK",  
 "message": "Backend is running",  
 "timestamp": "2025-07-23T18:30:00.000Z"  
}