**User Authentication System Backend Documentation**

**1. Introduction**

This document provides an overview of the authentication system implemented for my assignment. It includes explanations, code snippets, and testing details for key functionalities such as user registration, login, password reset, and security features.

Complete code here:

**2. Functional Modules**

**2.1 User Registration**

* Allows new users to create an account.
* Stores user details securely with password hashing.
* Sends a verification email upon successful registration.

**Register.php**

<?php

require\_once '../config/database.php';

require\_once '../utils/jwt.php';

require\_once '../utils/mailer.php';

$data = json\_decode(file\_get\_contents("php://input"));

if (!isset($data->username, $data->email, $data->password)) {

    die(json\_encode(["error" => "Missing required fields"]));

}

$username = $data->username;

$email = $data->email;

$password = password\_hash($data->password, PASSWORD\_BCRYPT);

$referral\_code = uniqid();

$referred\_by = isset($data->referral\_code) ? $data->referral\_code : NULL;

$subject = "Welcome to VomyChat!";

$message = "<h2>Thank you for signing up!</h2><p>Your account is now active.</p>";

if (sendMail($email, $subject, $message)) {

    echo "Email sent successfully!";

} else {

    echo "Failed to send email.";

}

$check\_sql = "SELECT id FROM users WHERE username = ? OR email = ?";

$check\_stmt = $conn->prepare($check\_sql);

$check\_stmt->execute([$username, $email]);

if ($check\_stmt->rowCount() > 0) {

    die(json\_encode(["error" => "Username or email already exists"]));

}

$sql = "INSERT INTO users (username, email, password\_hash, referral\_code, referred\_by) VALUES (?, ?, ?, ?, ?)";

$stmt = $conn->prepare($sql);

$stmt->execute([$username, $email, $password, $referral\_code, $referred\_by]);

$user\_id = $conn->lastInsertId();

if ($referred\_by) {

    $referrer\_id = $conn->prepare("SELECT id FROM users WHERE referral\_code = ?");

    $referrer\_id->execute([$referred\_by]);

    $referrer\_id = $referrer\_id->fetchColumn();

    if ($referrer\_id) {

        $sql = "INSERT INTO referrals (referrer\_id, referred\_user\_id, status) VALUES (?, ?, 'pending')";

        $conn->prepare($sql)->execute([$referrer\_id, $user\_id]);

    }

}

$referral\_link = "http://yourdomain.com/register?referral=" . $referral\_code;

echo json\_encode([

    "message" => "User registered successfully",

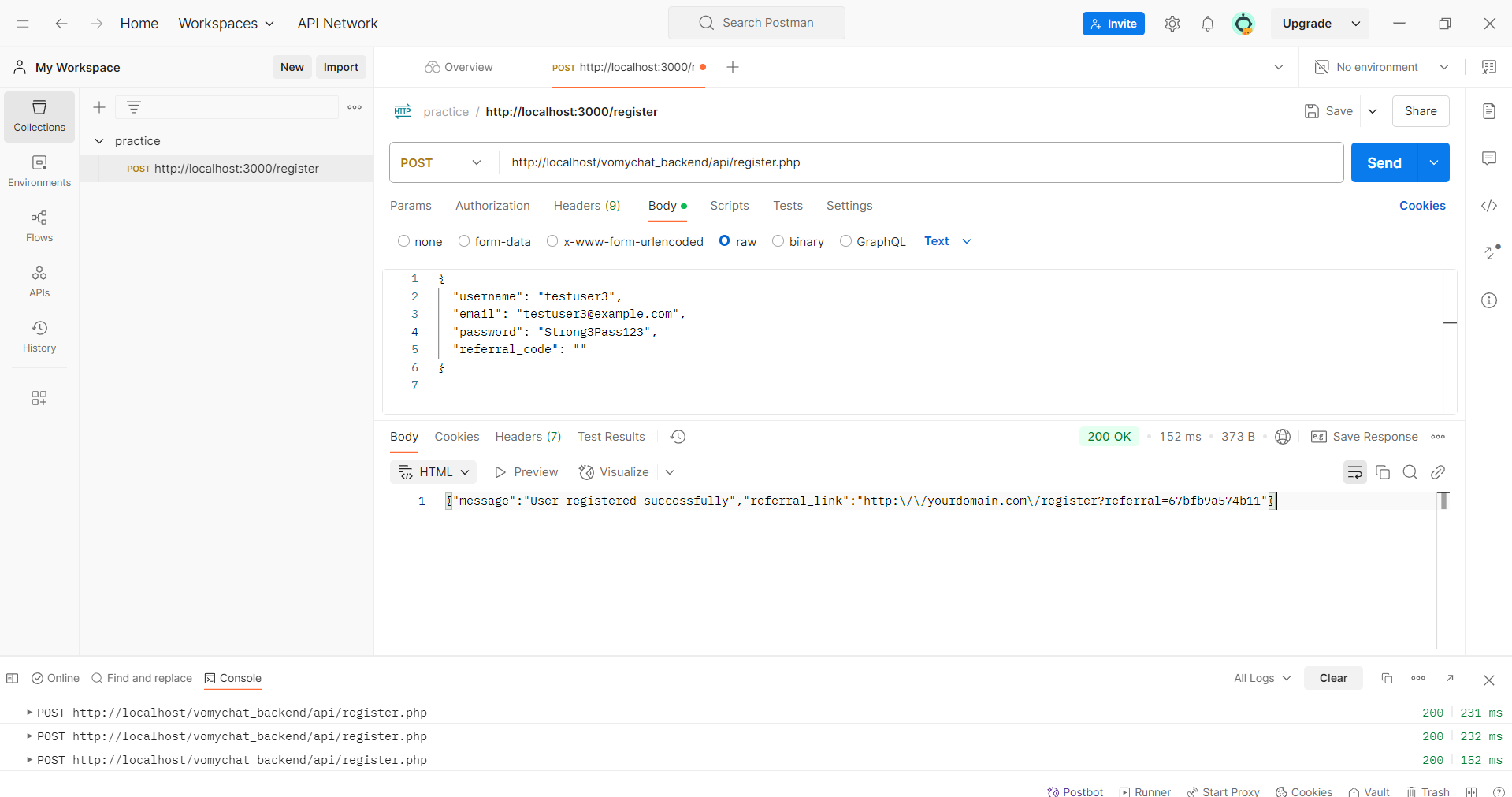
    "referral\_link" => $referral\_link

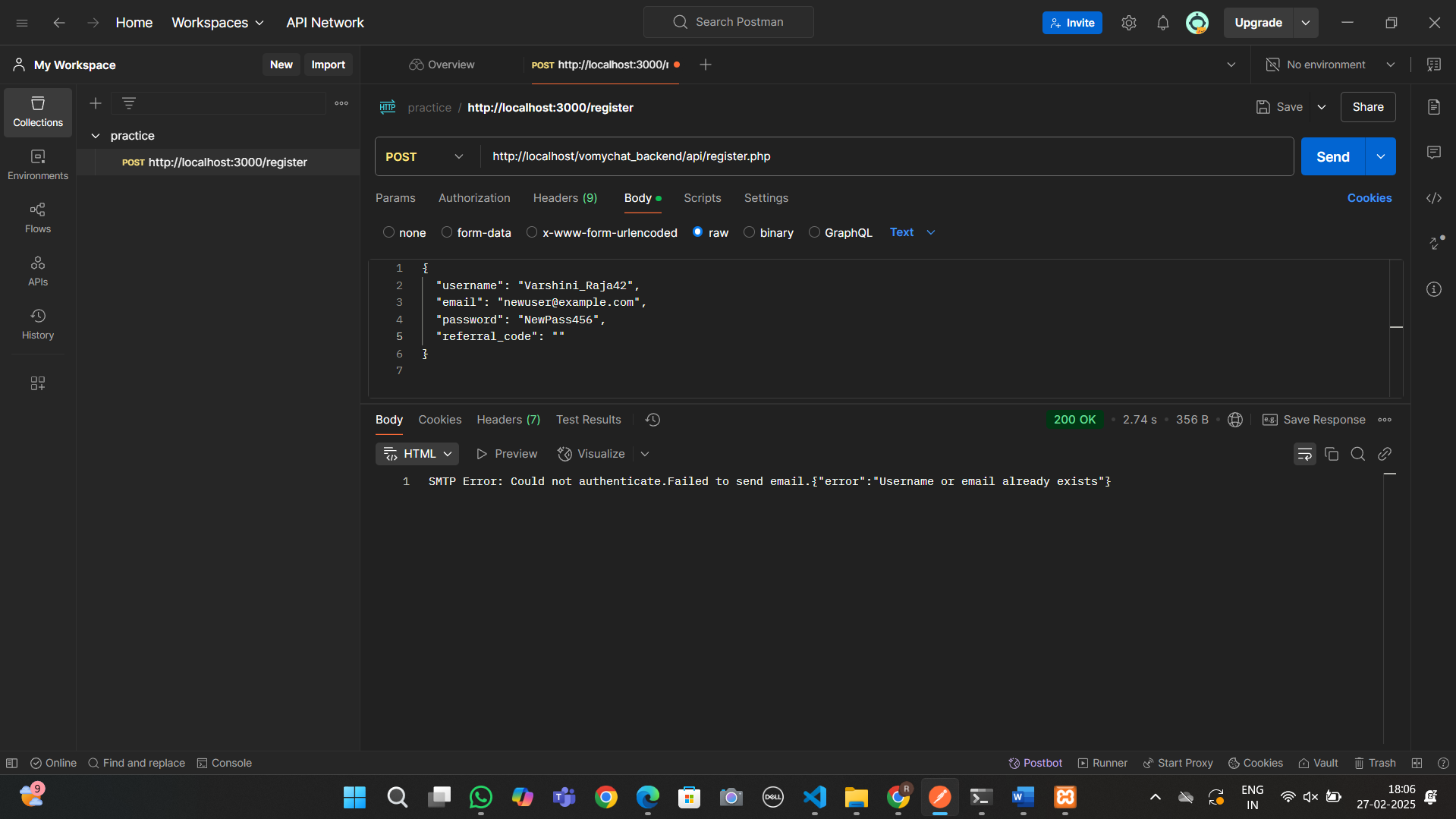
]);

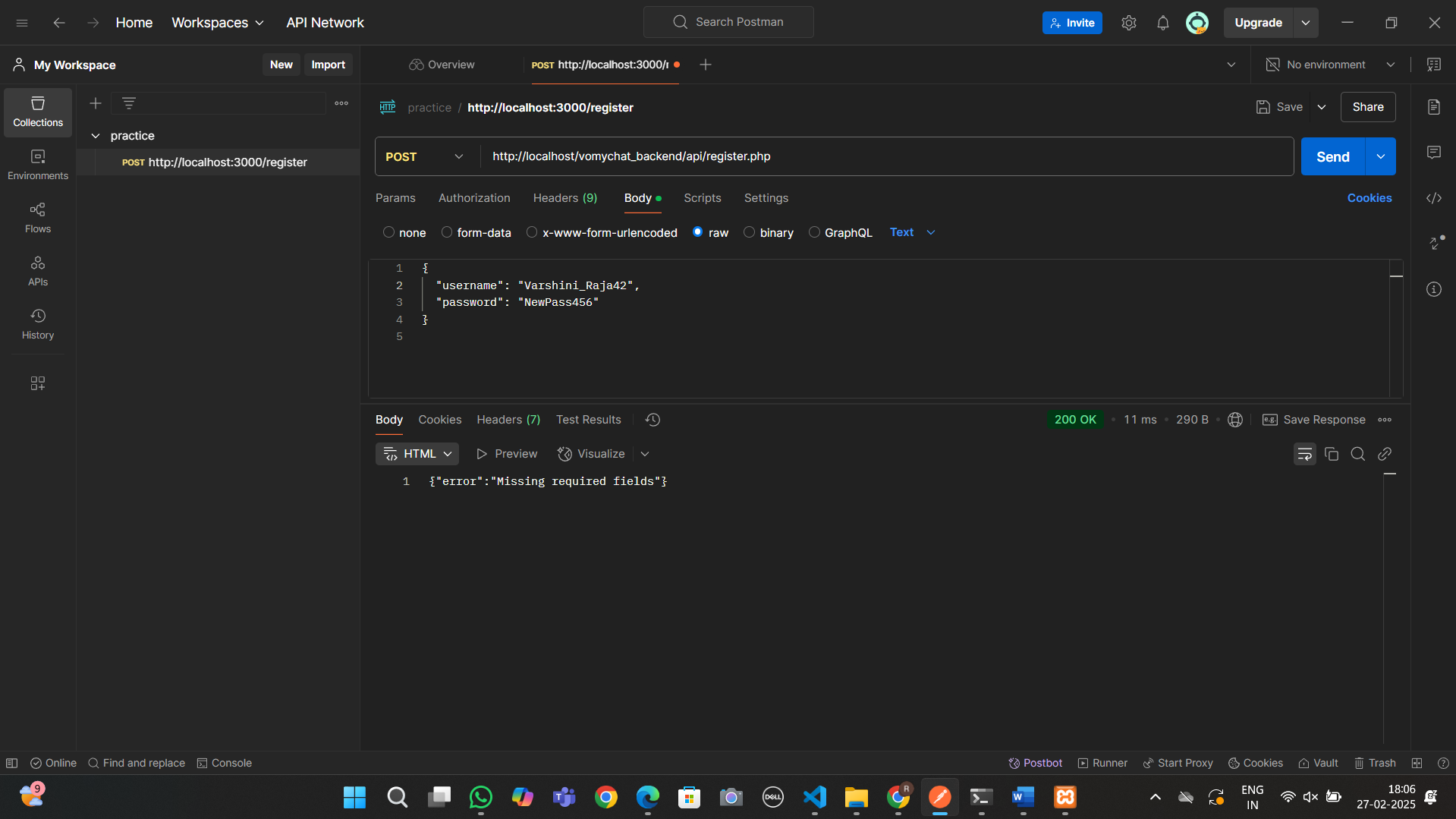
?>

**Test Cases:**

1. Register with valid details

  
2. Register with an existing email

  
3. Register without required fields



**2.2 User Login**

* Verifies user credentials.
* Generates a JWT token upon successful authentication.

**Code:**

<?php

require\_once '../config/database.php';

require\_once '../utils/jwt.php';

$data = json\_decode(file\_get\_contents("php://input"));

$email = $data->email;

$password = $data->password;

$stmt = $conn->prepare("SELECT \* FROM users WHERE email = ?");

$stmt->execute([$email]);

$user = $stmt->fetch(PDO::FETCH\_ASSOC);

if (!$user || !password\_verify($password, $user['password\_hash'])) {

    die(json\_encode(["error" => "Invalid credentials"]));

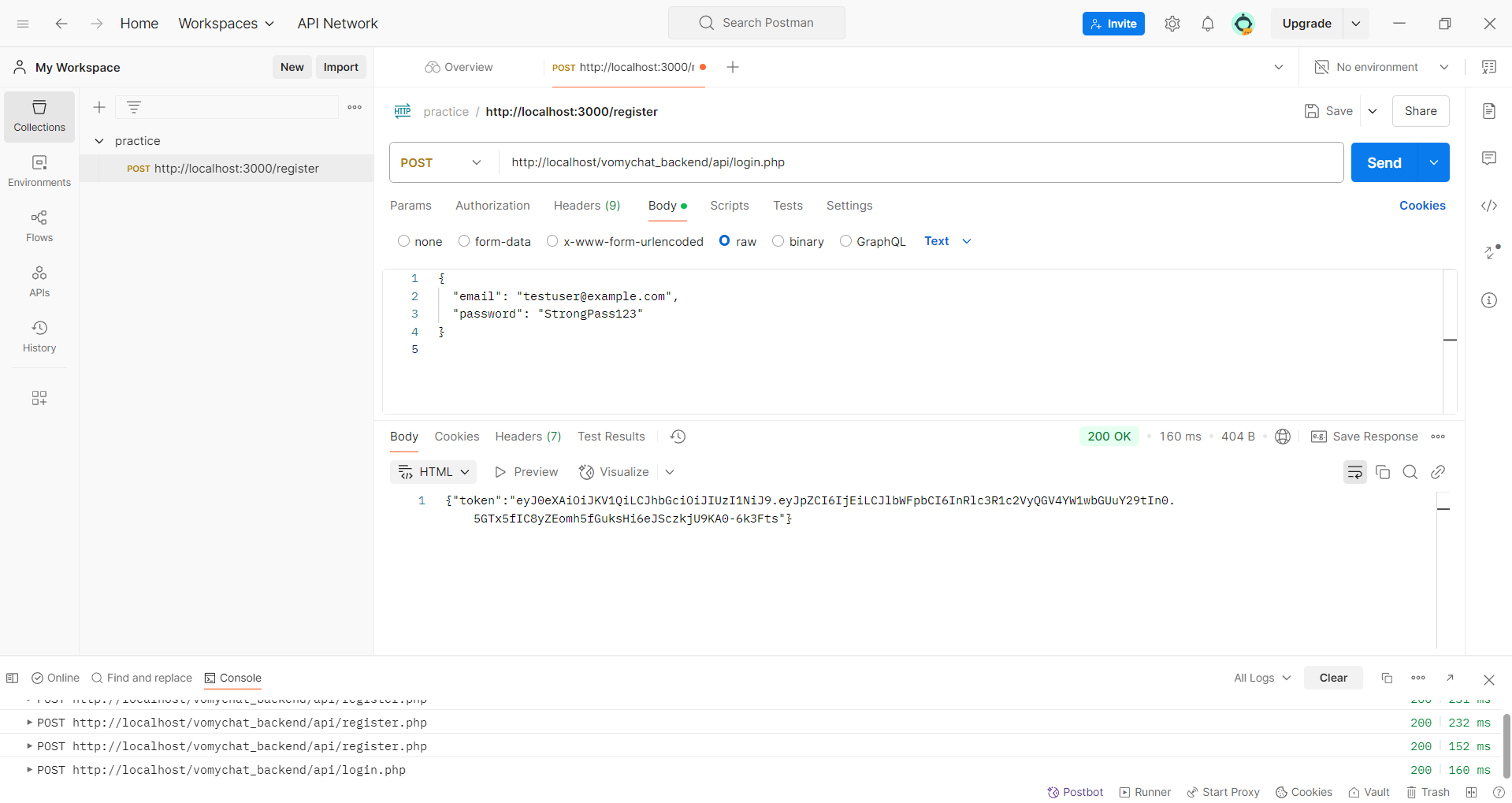
}

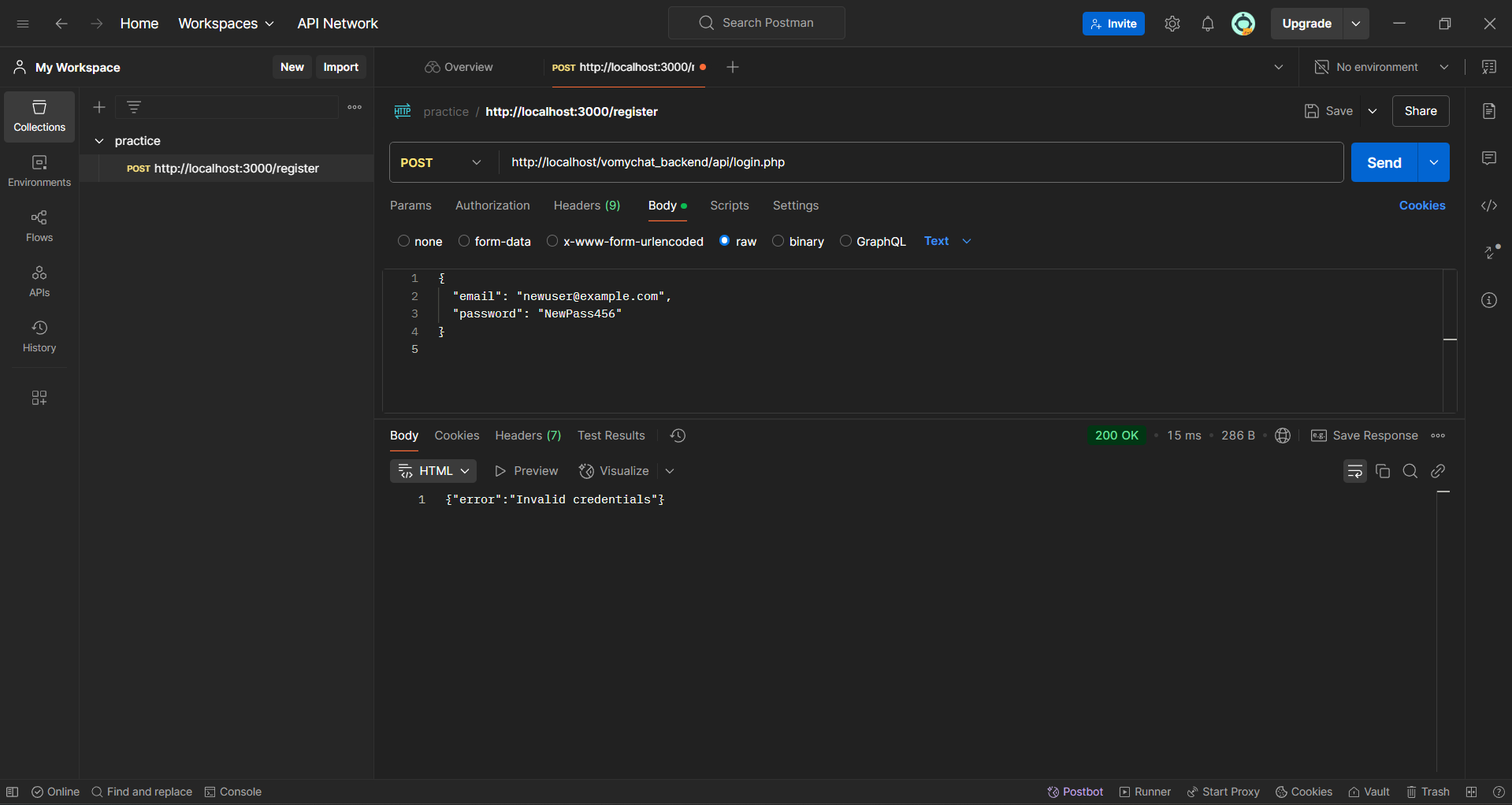
$token = generate\_jwt(["id" => $user['id'], "email" => $user['email']]);

echo json\_encode(["token" => $token]);

?>

**Test Cases:** ☐ Login with correct credentials

  
☐ Login with incorrect password



**2.3 Forgot Password**

* Generates a reset token and sends an email with a reset link.

**Code:**

<?php

require\_once '../config/database.php'; // Database connection

require\_once '../utils/mailer.php';

require\_once '../utils/jwt.php';

header('Content-Type: application/json');

$data = json\_decode(file\_get\_contents("php://input"), true);

if (!isset($data['email'])) {

    echo json\_encode(["error" => "Email is required"]);

    exit();

}

$email = $data['email'];

// Check if email exists in DB

$stmt = $conn->prepare("SELECT id FROM users WHERE email = ?");

$stmt->execute([$email]);

$user = $stmt->fetch();

if (!$user) {

    echo json\_encode(["error" => "No account found with this email"]);

    exit();

}

// Generate reset token (JWT)

$token = generate\_jwt(["id" => $user['id'], "email" => $email], 3600); // 1 hour expiry

// Reset link

$resetLink = "http://localhost/vomychat\_backend/api/reset\_password.php?token=$token";

// Email content

$subject = "Password Reset Request";

$message = "<h3>Reset Your Password</h3>

            <p>Click the link below to reset your password:</p>

            <a href='$resetLink'>$resetLink</a>

            <p>This link is valid for 1 hour.</p>";

if (sendMail($email, $subject, $message)) {

    echo json\_encode(["success" => "Reset link sent to email"]);

} else {

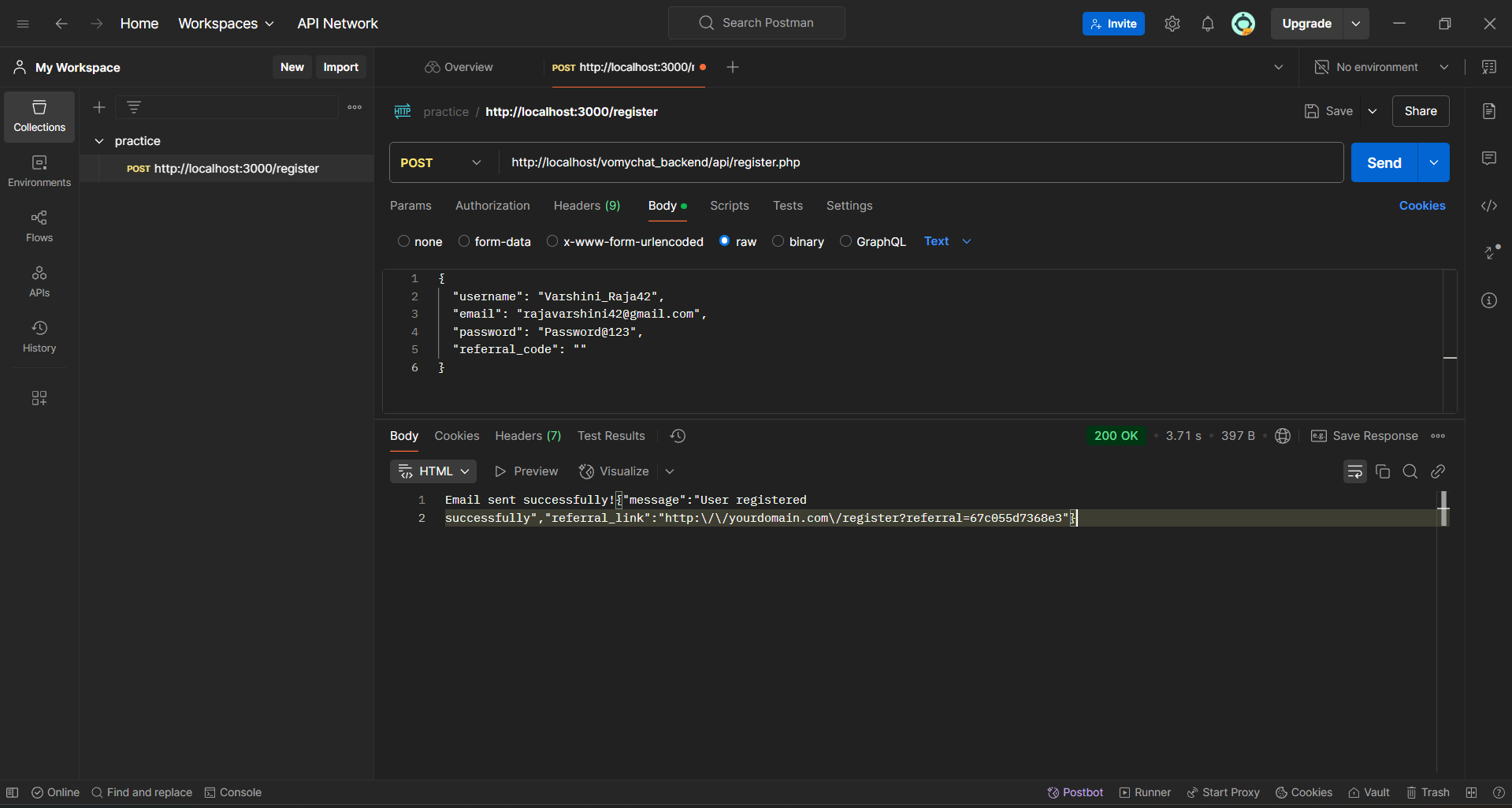
    echo json\_encode(["error" => "Failed to send email"]);

}

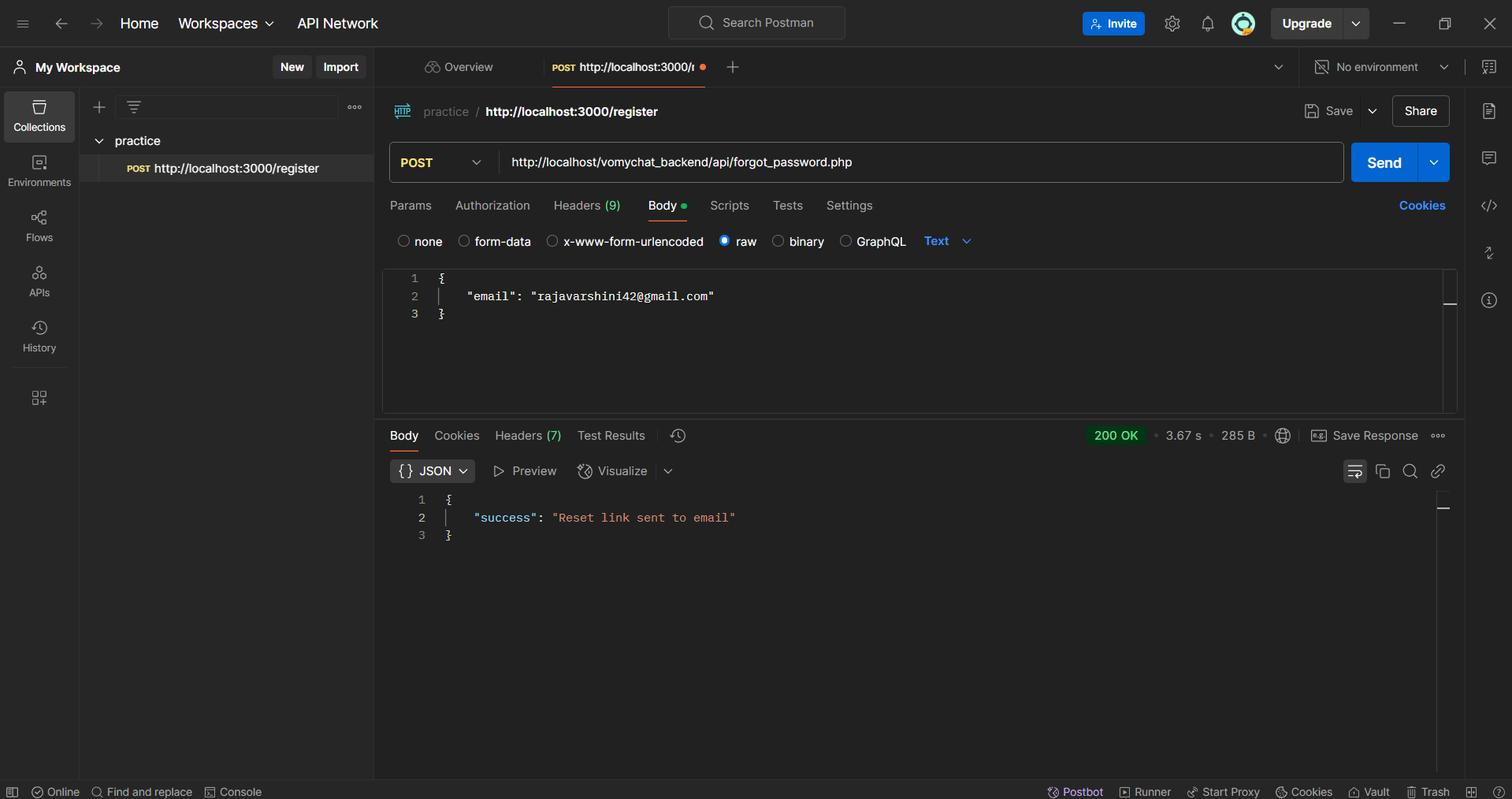
?>

**Test Cases:**

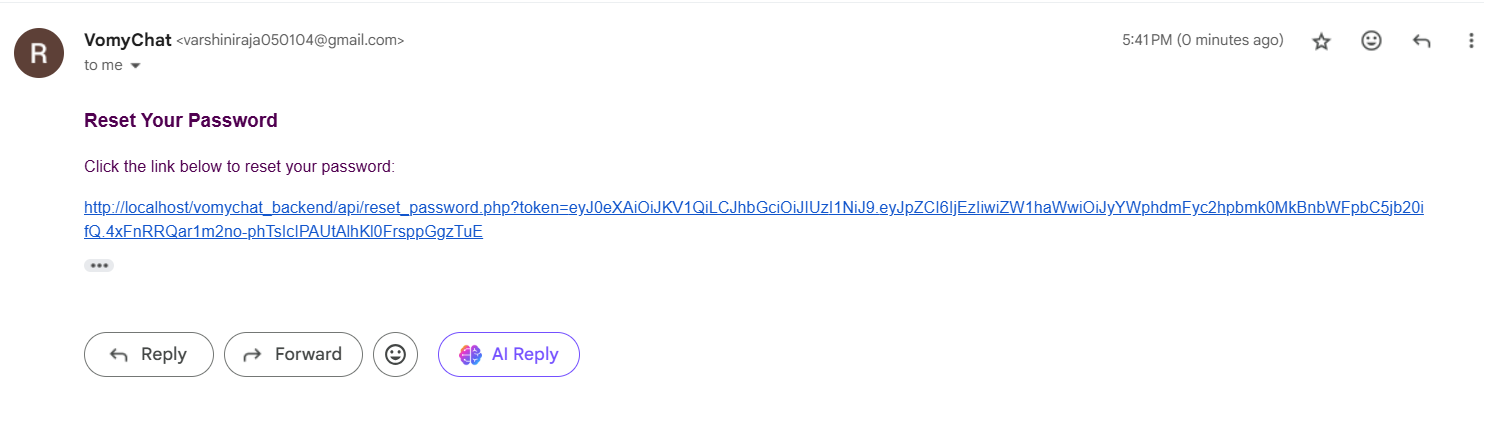
Registration of a new user:



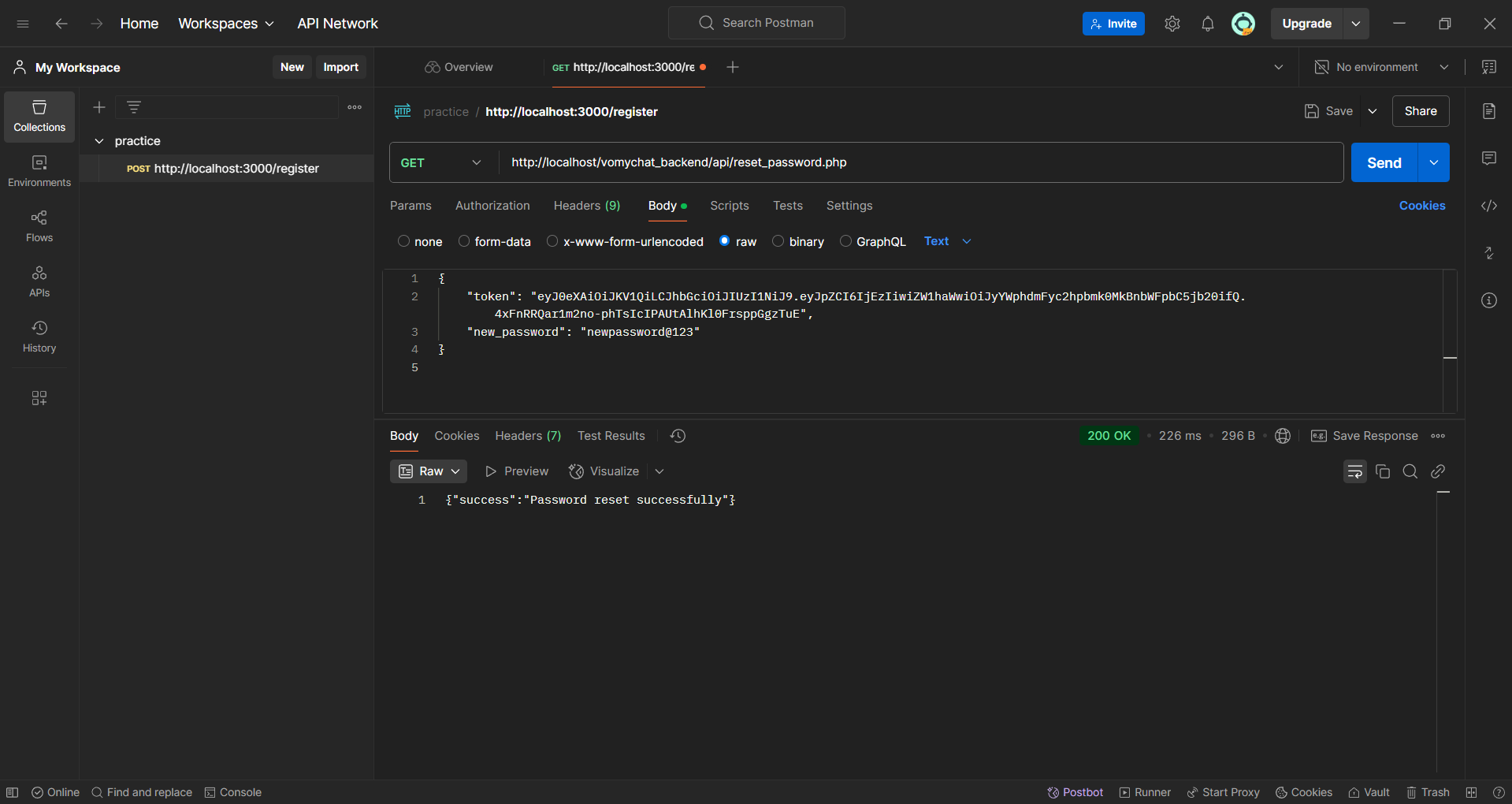
Welcoming Mail  


Requesting forgot\_password with email  


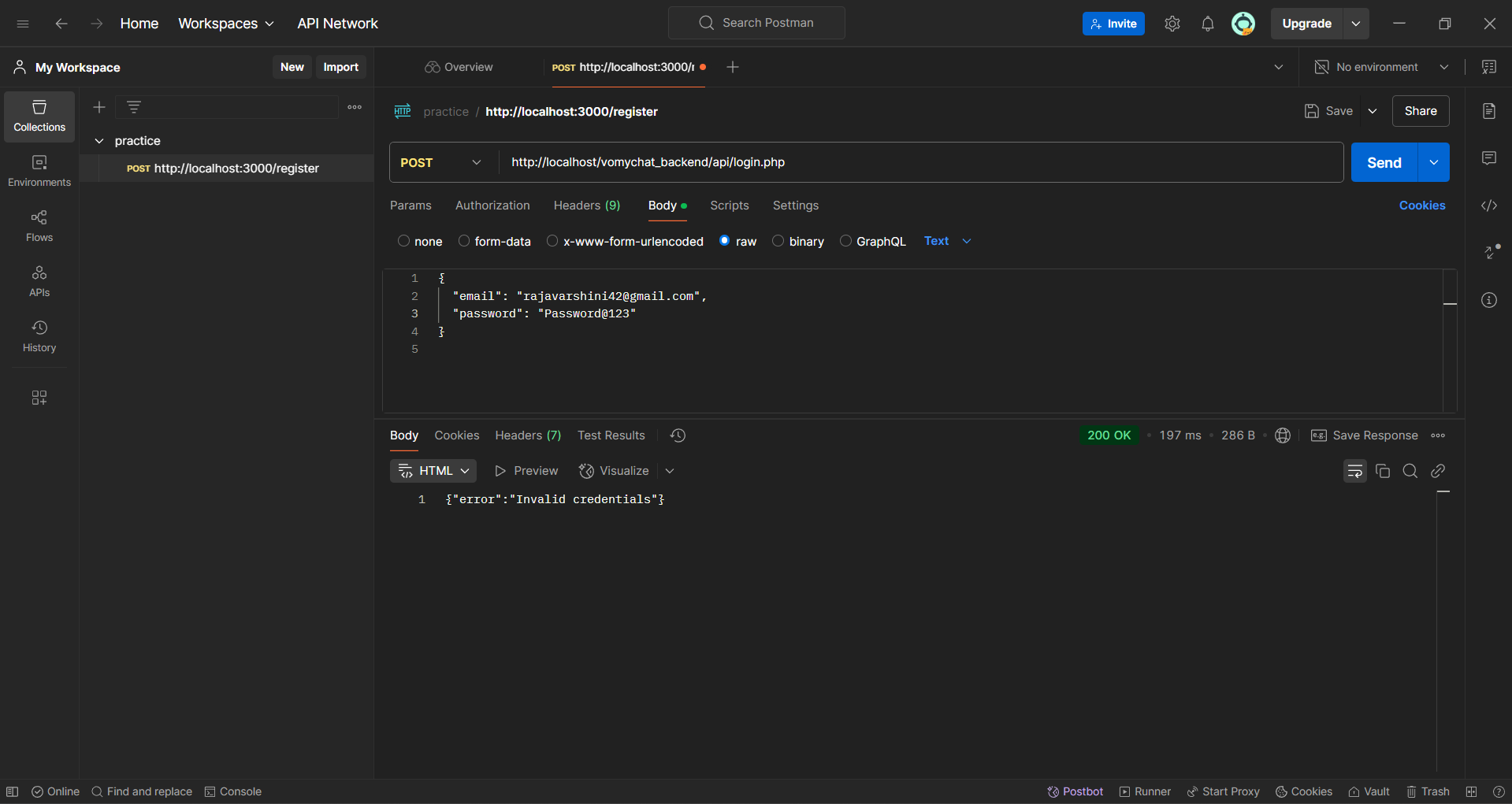
Reset link sent to email



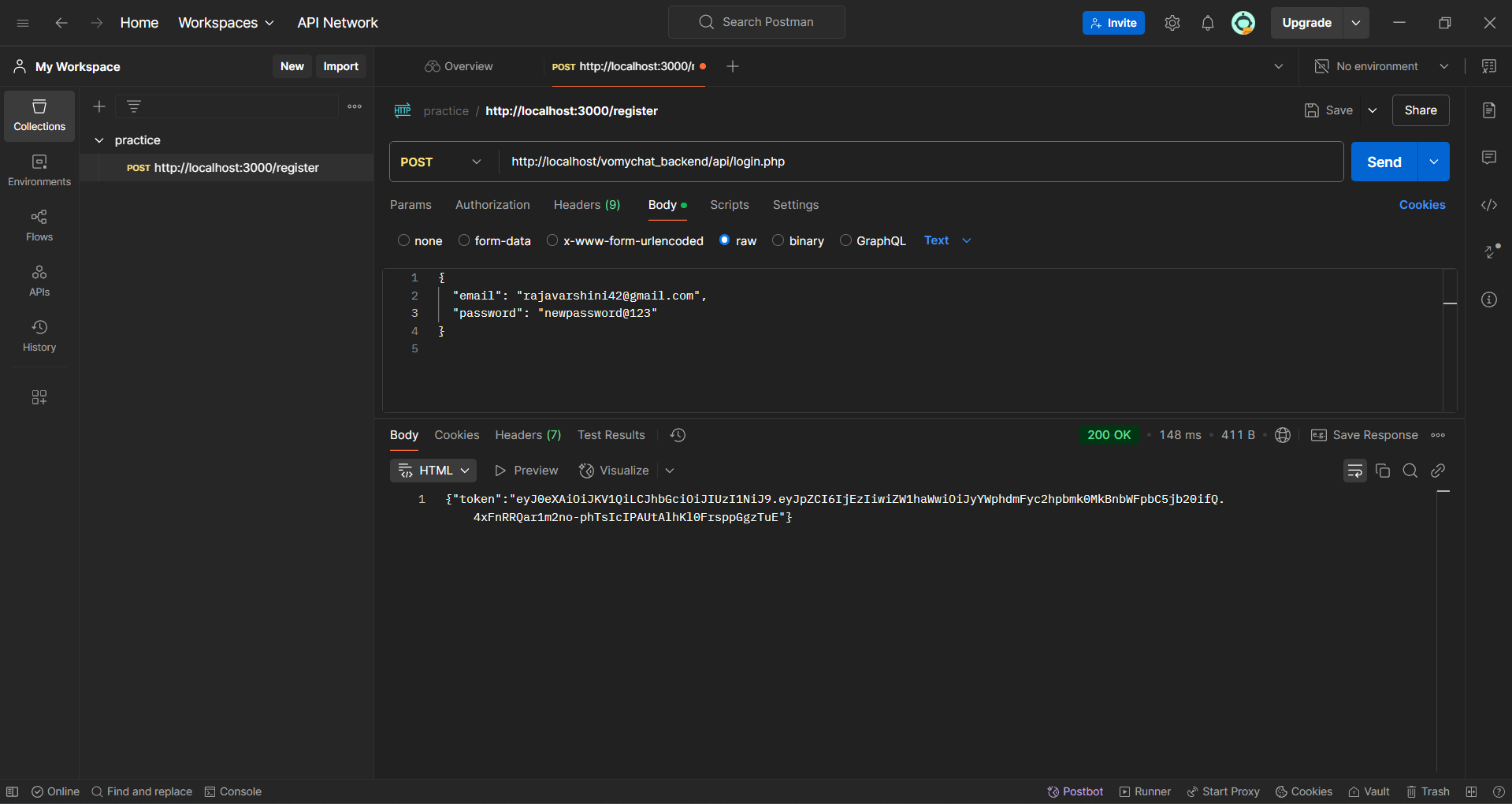
Trying to reset password using obtained token



Login using old password:



Login using new password:



**3. Security Considerations**

**3.1 Token Expiry & Validation**

* JWTs expire after one hour for security.
* Any tampering of the token results in rejection.

**3.2 SQL Injection Protection**

* All database queries use prepared statements.
* Input data is sanitized before execution.

**3.3 Brute Force Protection**

* Implement login attempt rate limiting.

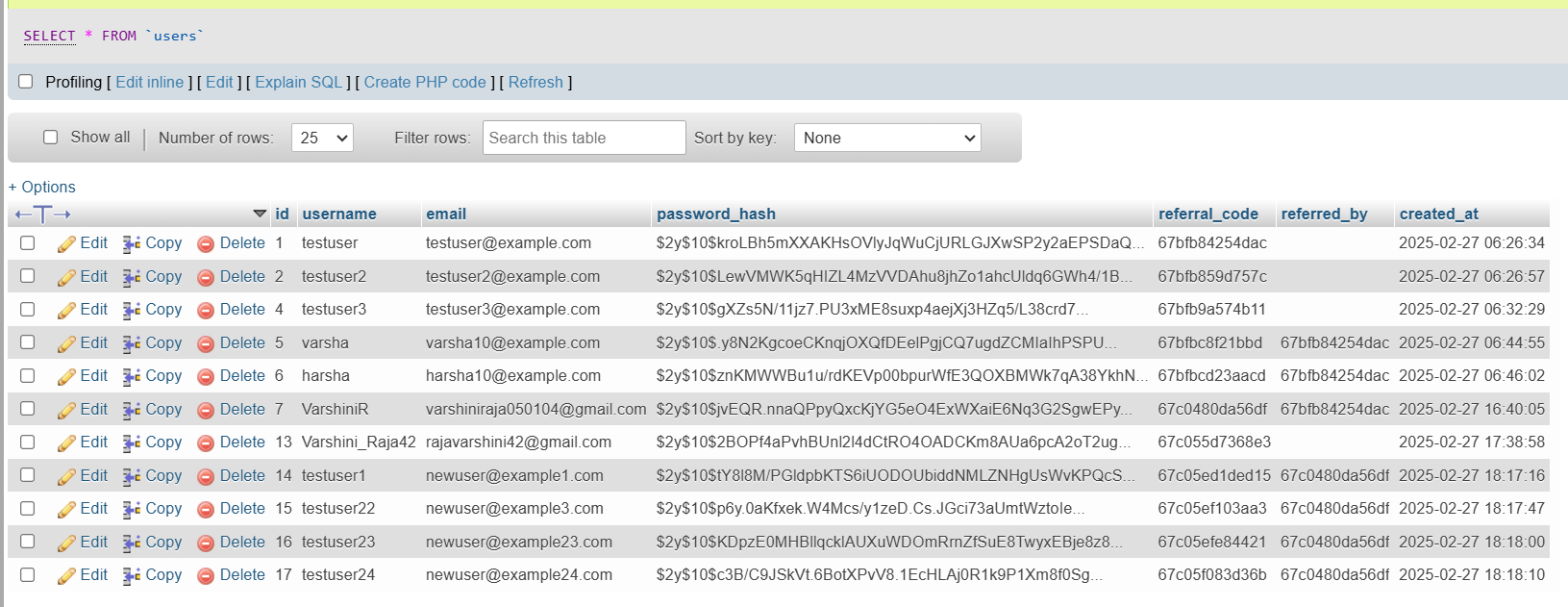
**4. Referral System**

The referral system in VomyChat allows users to invite others to join the platform. Each user gets a unique referral link that they can share, and when a new user registers using that link, the referrer gets credited. This helps grow the user base while rewarding existing users.

Implementation Details

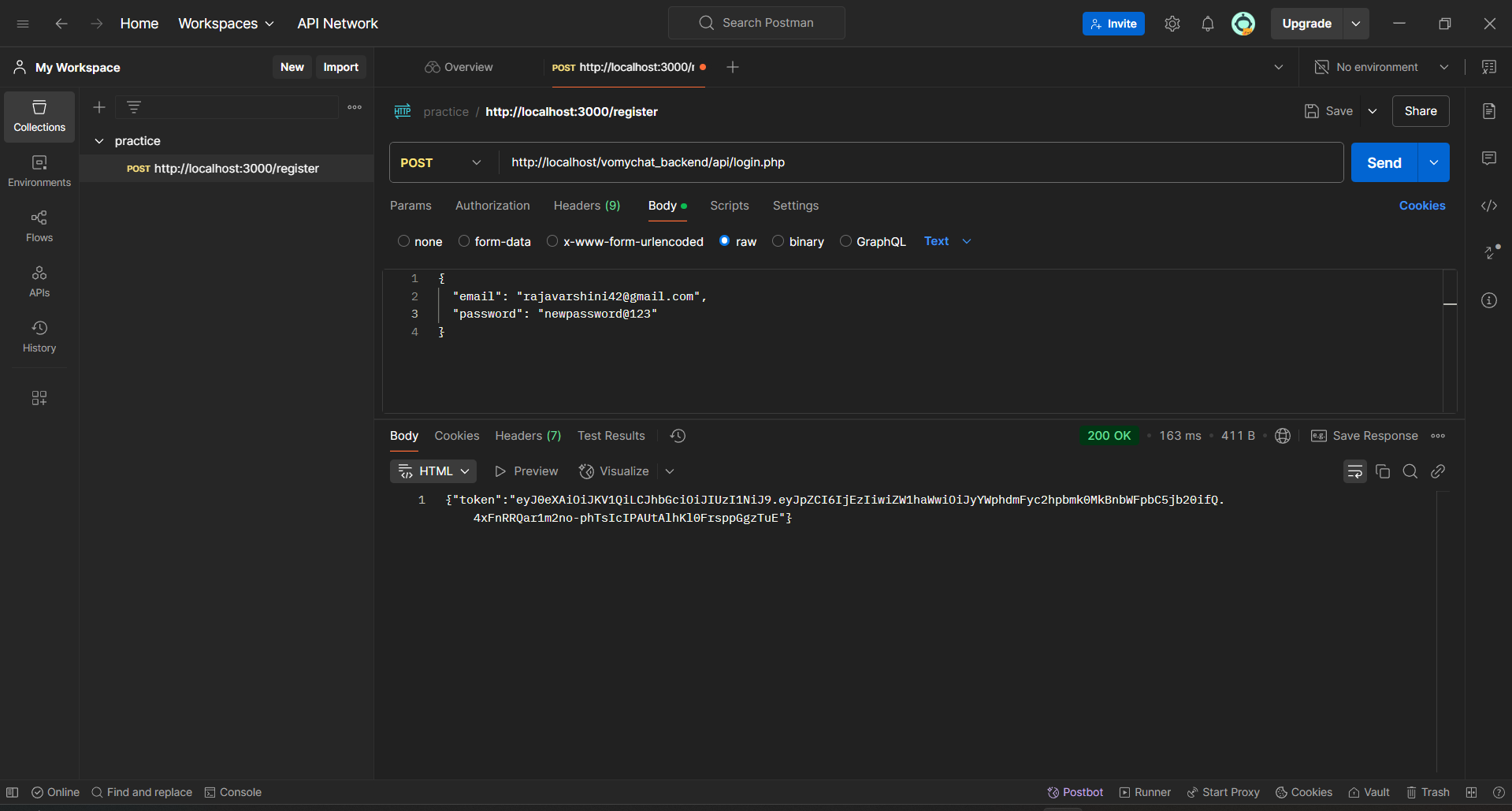
1. Generating Referral Links
   * A unique referral code is created for each user upon registration.
   * This code is used to generate a referral link.
2. Tracking Referrals
   * When a new user registers using a referral link, the system links their account to the referring user.
   * The number of referrals is stored in the database.
3. Viewing Referral Statistics
   * Users can see how many people they have referred and whether those referrals are active**.**

**The Users table containing all user details in XAMPP’s phpMyAdmin**

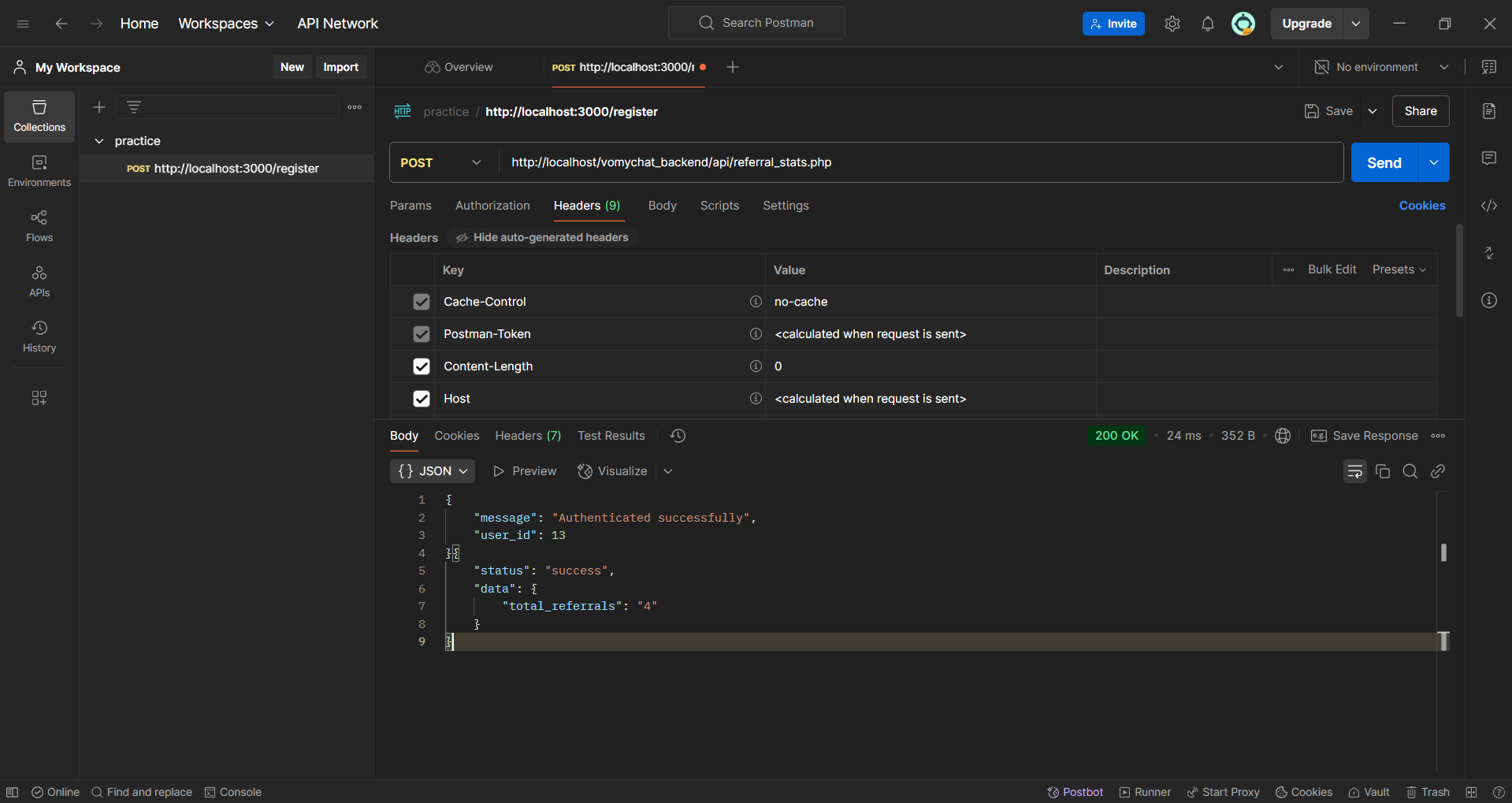
****

**Fetching referral statistics of a particular user:**

1. **Logging in to get the token**

****

1. **Using token in the headers to get referral statistics**

****

This document outlines the implementation and testing of the user authentication system. Each module has been tested for functionality, security, and robustness. Screenshots will be attached to verify successful execution.