EmailJS Setup Guide for Code Analyzer

This guide walks you through setting up email functionality for the Enhanced Code Analyzer using EmailJS.

Step 1: Create EmailJS Account

1. Go to EmailJS.com
2. Sign up for a free account
3. Verify your email address

Step 2: Create Email Service

1. In your EmailJS dashboard, go to **Email Services**
2. Click **Add New Service**
3. Choose your email provider (Gmail, Outlook, etc.)
4. Follow the setup instructions for your provider
5. Note down your **Service ID** (e.g., `service_abc123`)

Step 3: Create Email Template

1. Go to **Email Templates** in your dashboard
2. Click **Create New Template**
3. Use this template structure:

Template Content:

Subject: Code Analysis Results - {{analysis_date}}

Body:

```
Hello,

Your code analysis is complete! Here are the results:

Analysis Summary:
- Date: {{analysis_date}}
- Total Files Analyzed: {{total_files}}
- Total Issues Found: {{total_issues}}

Detailed Results:
{{message}}

---
This email was generated by Enhanced Code Analyzer.
Generated on {{analysis_date}}
```

Template Variables:

- `{{to_email}}` - Recipient email
- `{{from_name}}` - Sender name
- `{{subject}}` - Email subject
- `{{message}}` - Analysis results
- `{{analysis_date}}` - Analysis timestamp
- `{{total_files}}` - Number of files analyzed
- `{{total_issues}}` - Total issues found

4. Save the template and note the **Template ID** (e.g., `template_xyz789`)

Step 4: Get Your User ID

1. Go to **Account** → **General**
2. Find your **User ID** (e.g., `user_abc123`)

Step 5: Configure the Code

Update these values in the code:

```
javascript
```

```
// In the initializeEmailJS function
window.emailjs.init("YOUR_USER_ID_HERE"); // Replace with your User ID

// In the sendAnalysisEmail function
await window.emailjs.send(
  'YOUR_SERVICE_ID_HERE',  // Replace with your Service ID
  'YOUR_TEMPLATE_ID_HERE', // Replace with your Template ID
  templateParams
);
```

Step 6: Update the Code

Replace these placeholders in the `enhanced_code_analyzer.js`:

1. **Line with `YOUR_EMAILJS_USER_ID`** → Your actual User ID
2. **Line with `YOUR_EMAIL_SERVICE_ID`** → Your actual Service ID
3. **Line with `YOUR_EMAIL_TEMPLATE_ID`** → Your actual Template ID

Example:

```
javascript

// Before
window.emailjs.init("YOUR_EMAILJS_USER_ID");

// After
window.emailjs.init("user_abc123");
```

Step 7: Test the Setup

1. Enter a valid email address in the app
2. Check "Email results to me when analysis is complete"
3. Run a code analysis
4. Check your email for the results

Troubleshooting

Common Issues:

Email not sending:

- Check your Service ID, Template ID, and User ID are correct

- Verify your email service is properly connected in EmailJS dashboard
- Check browser console for error messages

Template variables not working:

- Ensure variable names in template match exactly: `{{analysis_date}}`, `{{message}}`, etc.
- Variables are case-sensitive

Gmail/Outlook issues:

- Enable "Less secure app access" for Gmail (or use App Passwords)
- For Outlook, ensure you're using the correct authentication method

EmailJS Limits (Free Plan):

- 200 emails per month
- Email templates must be under 50KB
- Rate limit: 1 email per second

Alternative Email Services

If you need more email capacity, consider:

1. **SendGrid** - More robust for high-volume
2. **Mailgun** - Developer-friendly email API
3. **AWS SES** - Cost-effective for large volumes
4. **Custom Backend** - Full control with Node.js/Express

Security Notes

- Never expose your EmailJS private key in frontend code
- EmailJS public keys are safe to use in client-side applications
- Consider implementing rate limiting for production use
- Validate email addresses before sending

Email Content Customization

You can customize the email format by modifying the `formatResultsForEmail` function:

```
javascript
```

```
const formatResultsForEmail = (results) => {
  // Add custom formatting here
  // Include charts, links, or styled content
  // Return formatted string
};
```

Production Considerations

For production deployment:

1. Set up environment variables for EmailJS keys
2. Implement proper error handling and user feedback
3. Add email validation and sanitization
4. Consider implementing email queuing for large analyses
5. Monitor EmailJS usage and upgrade plan if needed

Once configured, users will receive professional-looking email reports with their code analysis results, making it easy to share findings with team members or keep records of code quality assessments.