Email Server — Integration Guide

outgoingEmail Schema · API Reference · Project: faxserver-19efb

Send an email from any app by POSTing a JSON payload to the API endpoint or by writing directly to the **outgoingEmail** Firestore collection. The **sendEmail** Cloud Function handles delivery automatically within seconds.

1. API Endpoint

Production URL (custom domain):

```
POST  https://emailserver.jeavensoft.com/queueEmail
```

Cloud Function URL (fallback):

```
POST  https://us-central1-faxserver-19efb.cloudfunctions.net/queueEmail
```

Header required: `Content-Type: application/json`

2. Download Helper Files

Drop one of these into your project — no SDK or dependencies needed.

File	Use in	Download
`sendEmail.php`	PHP apps — eGarage, POS, invoicing, etc.	Download sendEmail.php
`sendEmail.js`	JavaScript / Node.js / browser apps	Download sendEmail.js

3. Sample JSON Payload

Minimum (required fields only):

```
JSON

{

  "toEmail":  "customer@example.com",

  "subject":  "Invoice #123",

  "message":  "Hi John, your invoice is ready."

}
```

Full payload (all fields + custom fields):

```
JSON
```

```
{

  "fromEmail":      "info@jeavensoft.com",    // optional — defaults to
mail@jeavensoft.com

  "fromName":       "Jeavensoft Invoices",    // optional — display name in inbox

  "toEmail":        "customer@example.com",   // REQUIRED

  "toName":         "John Smith",             // optional

  "subject":        "Invoice #123",           // REQUIRED

  "message":        "Hi John,\n\nYour invoice is attached.",  // REQUIRED

  "replyEmail":     "support@jeavensoft.com", // optional — reply-to address

  "attachmentPath": "https://firebasestorage.googleapis.com/...",  // optional

  "attachmentName": "invoice-123.pdf",        // optional (required if attachmentPath set)


  // Any extra fields are saved to the doc as-is:

  "invoiceId":  "INV-123",

  "clientId":   "C001",

  "appName":    "eGarage"

}
```

4. PHP Usage

```
PHP

<?php

require 'sendEmail.php';  // download from link above


$result = sendEmail([

    'toEmail'   => 'customer@example.com',

    'subject'   => 'Invoice #123',

    'message'   => 'Hi, your invoice is ready.',

    'fromEmail' => 'info@jeavensoft.com',  // optional

    'invoiceId' => 'INV-123',              // extra fields — saved to doc

    'appName'   => 'eGarage',
```

```
]);

if ($result['success']) {
    echo 'Queued! Doc ID: ' . $result['docId'];
} else {
    echo 'Error: ' . $result['error'];
}
```

5. JavaScript / AJAX Usage

```
JavaScript
import { sendEmail } from './sendEmail.js';  // download from link above

const result = await sendEmail({
    toEmail:   'customer@example.com',
    subject:   'Invoice #123',
    message:   'Hi, your invoice is ready.',
    fromEmail: 'info@jeavensoft.com',  // optional
    invoiceId: 'INV-123',              // extra fields — saved to doc
    appName:   'eGarage',
});

if (result.success) {
    console.log('Queued! Doc ID:', result.docId);
} else {
    console.error('Error:', result.error);
}
```

6. From Email Accounts — SMTP Setup

Before sending email, each *fromEmail* address must have an entry in the **emailFromAccounts** Firestore collection with valid SMTP credentials. Add accounts via the **remoteemail.web.app** admin console or write directly to Firestore.

emailFromAccounts document structure:

```
Firestore / JSON

{

  "fromEmail":   "info@jeavensoft.com",   // REQUIRED — the from address

  "displayName": "Jeavensoft",            // shown as sender name in inbox

  "smtpHost":    "email-smtp.us-east-1.amazonaws.com",  // REQUIRED

  "smtpPort":    587,                     // REQUIRED — 587 (STARTTLS) or 465 (SSL)

  "smtpUser":    "AKIA6QCB5ZKQNBV7B6WQ", // REQUIRED — SMTP username

  "smtpPass":    "••••••••••••••••",      // REQUIRED — SMTP password / App Password

  "provider":    "AWS SES"                // informational

}
```

SMTP settings by provider — auto-populated on remoteemail.web.app:

Provider	smtpHost	Port	Password / Credential
AWS SES (own domain)	`email-smtp.us-east-1.amazonaws.com`	587	SES SMTP credentials (AKIA… username + generated password from SES console)
Gmail (@gmail.com)	`smtp.gmail.com`	587	Google App Password — NOT your Gmail password. Enable 2FA first, then Google Account → Security → App Passwords
Yahoo (@yahoo.com)	`smtp.mail.yahoo.com`	587	Yahoo App Password — Yahoo Account Security → Generate app password
Outlook / Hotmail	`smtp-mail.outlook.com`	587	Microsoft account password, or App Password if 2FA is enabled
Custom domain (hosting)	`mail.yourdomain.com`	587	Your hosting email account password. Get SMTP host from your hosting control panel

Important: If *fromEmail* is left blank when calling the API, the system automatically uses `mail@jeavensoft.com` as the default sender. Ensure this account exists in *emailFromAccounts* with valid AWS SES credentials.

Default fallback address: `mail@jeavensoft.com` (AWS SES — domain jeavensoft.com verified in SES)

6. API Response

Success:

```
{ "success": true,  "docId": "26uRTUT2Slyksyz5W9NP" }
```

Error:

```
{ "success": false, "error": "Missing required field(s): toEmail, subject" }
```

7. Field Reference

Field	Required	Type	Notes
`toEmail`	**YES**	`string`	Recipient email address
`subject`	**YES**	`string`	Email subject line
`message`	**YES**	`string`	Plain text — \n converted to in HTML
`fromEmail`	no	`string`	Defaults to *mail@jeavensoft.com* if omitted
`fromName`	no	`string`	Display name shown in recipient inbox
`toName`	no	`string`	Recipient display name in To: header
`replyEmail`	no	`string`	Reply-To address — defaults to fromEmail
`attachmentPath`	no	`string`	Firebase Storage download URL
`attachmentName`	no	`string`	Filename shown to recipient
`...extras`	no	`any`	Any extra fields are saved to the Firestore doc

8. Fields Written Back Automatically

Do **not** set these — they are written by the Cloud Functions after sending.

Field	Written by	Values / Notes
`status`	`sendEmail`	new → processing → delivered / fail / error
`statusMessage`	`sendEmail`	SMTP response e.g. '250 OK'
`messageId`	`sendEmail`	SES message ID
`sentAt / failedAt`	`sendEmail`	Timestamps
`openCount`	`trackOpen`	Human (non-bot) open count
`openedAt / lastOpenAt`	`trackOpen`	First and last open timestamps

sesStatus	trackSESEvent	delivered / bounced / soft-bounce / complained / rejected
sesDeliveredAt	trackSESEvent	Confirmed delivery timestamp
sesBounceType	trackSESEvent	Permanent / Transient
sesDiagnosticCode	trackSESEvent	e.g. 550 user unknown

9. Firestore Collections

Collection	Purpose
outgoingEmail	Every email queued for sending — the main queue
emailFromAccounts	SMTP account configs — one doc per sender
emailTracker	Per-sender sent count + last sent
emailOpenTracker	Open pixel events (1 per 60s window per email)
emailSESEventLog	Raw SES delivery / bounce / complaint events
emailServerInfo	Monthly sent totals — emailServerInfo/aws/monthly/YYYY-MM