

ST Payment Pages Setup Guide

Version 1.3

Copyright

© SecureTrading 2011. All rights reserved. No part of this document may be photocopied, reproduced, stored in a retrieval system or transmitted in any form or by any means whether electronic, mechanical or otherwise without the prior written permission of SecureTrading Ltd.

Disclaimer

This document is for informational purposes only. SecureTrading make no warranties, express or implied, through the distribution of this document. No warranty of accuracy is given concerning the contents of the information contained in this publication. To the extent permitted by law no liability (including liability to any person by reason of negligence) will be accepted by SecureTrading, its subsidiaries or employees for any direct or indirect loss or damage caused by omissions from or inaccuracies in this document.

SecureTrading reserves the right to revise the content without obligation to notify any person of such changes.

Document revised on 31-Mar-2011.

Contents

1	Introduction.....	4
1.1	Your sitereference.....	4
1.2	Test mode.....	4
1.3	Callback.....	4
1.4	Security.....	5
	Quick start.....	7
1.5	Download the template files.....	7
1.6	Customise the order page.....	7
1.7	Testing.....	8
1.7.1	Successful transaction.....	8
1.7.2	Unsuccessful transaction.....	8
1.7.3	Further testing.....	9
2	Advanced customisation.....	10
2.1	The customisable files.....	10
2.2	The transaction process.....	10
2.3	Fields and variables.....	12
2.3.1	Fields.....	12
2.3.2	Essential fields.....	13
2.3.3	Variables.....	13
2.3.4	User defined fields and variables.....	14
2.4	Logging.....	14
2.5	Customising individual files.....	15
2.5.1	orderpage.html.....	15
2.5.2	form.html.....	15
2.5.3	success.html.....	16
2.5.4	failure.html.....	16
2.5.5	merchantemail.txt.....	17
2.5.6	customeremail.txt.....	17
2.5.7	failureemail.txt.....	17
2.5.8	UTF-8 email character encoding.....	17
2.6	Images in customised web pages.....	18
2.7	Multiple payment pages.....	18
2.8	Uploading files.....	19
3	Example using custom fields.....	20
4	Reference.....	22
4.1	Files.....	22
4.2	Fields and variables.....	23
4.3	Control fields.....	25
4.4	User defined fields.....	25
4.5	Reserved field names.....	25
4.6	Mandatory fields.....	26
4.7	Start/expiry date fields.....	26
4.8	Country codes.....	26
5	Redundancy planning.....	27
6	Appendix 1.....	29

1 Introduction

This guide explains how to configure your website to work with our SecureTrading Payment pages service. If you are using a commercial shopping cart system on your web site, then some of the contents of this guide might not apply.

See <http://www.securetrading.com/shopping-carts.html> on our web site for information on setting up particular shopping carts.

You need to have a SecureTrading account before you can set up your system; for instructions on how to obtain one, see our web page at: <http://www.securetrading.com/easysteps1.html> or contact our sales team on 0800 028 9151.

1.1 Your site reference

You should have received a “Welcome” email when your SecureTrading account was set up. This contains your **site reference**. Your site reference is important: you will need to quote it in ALL correspondence with SecureTrading, so please make it available to any of your staff who are likely to contact us (such as your developers, or other technical staff). Also, you will need your site reference to configure your web site correctly.

1.2 Test mode

Initially, your SecureTrading account will be set to **test mode**. In test mode, transactions are passed to the SecureTrading Test Bank, and **not** to the merchant’s acquiring bank. The SecureTrading Test Bank is a “dummy” bank set up to behave like the real banking networks, so although the correct responses are generated, no funds are transferred. You will not be charged for any transactions made while you are in test mode.

1.3 Callback

There are some references to our **callback** facility in this guide. By default, the SecureTrading system shows a success or failure web page, and sends emails to the merchant and customer after a credit card check. If you use **callback**, you can, in addition, run a script on your merchant web server. The script can make use of all the information (except credit card numbers) entered by your customers, and so can be used to keep detailed logs, update inventories, etc. For further information, see the **Callback Guide**, which you can download from <http://www.securetrading.com/support/general-setup-guides.html>

1.4 Security

To ensure that the data you have passed to SecureTrading has not been modified, you can supply a security field called `st_sitesecurity` that contains an md5 (cryptographic hash) of a predefined set of field values.

This option is disabled by default but can be easily enabled by following these steps below.

Firstly you will need to choose the fields values you are going to include for the hash, for this example the fields I will use are merchant, currency and the amount fields.

Once you have chosen your fields and the order in which you are going to use them you will also need to produce a password to include within your hash and also supply these details to SecureTrading so that we can produce a matching hash otherwise the transaction will be stopped.

Field Name	Field Value	Position
merchant	mysite1234	1 st
currency	GBP	2 nd
amount	1040	3 rd

Don't forget to include the password within the string to hash, the password should always be placed at the end of the string.

How to build the hash

Python Example

```
import md5
md5Obj = md5.new()
merchantFieldData = 'mysite1234GBP1040PASSWORD'
md5Obj.update(merchantFieldData)
merchantHash = md5Obj.hexdigest()
```

PHP Example

```
<?php
$merchantFieldData = 'mysite1234GBP1040PASSWORD';
$merchantHash = md5($merchantFieldData);
echo $merchantHash;
?>
```

Java Example

```
import java.math.*;
import java.lang.*;
import java.security.*;

public class mymd5 {
    public static void main(String args[]) throws Exception{
        String stringToHash="mysite1234GBP1040PASSWORD";
        MessageDigest mDigest=MessageDigest.getInstance("MD5");
        mDigest.update(stringToHash.getBytes(),0,stringToHash.length());
        String merchantHash=""+new BigInteger(1,mDigest.digest()).toString(16));
        System.out.println(merchantHash);
    }
}
```

Perl Example

```
use Digest::MD5 qw(md5 md5_hex);
$merchantFieldData = 'mysite1234GBP1040PASSWORD';
$merchantHash = md5_hex($merchantFieldData);
```

Now that the hashed string has been created the payment pages will need to be modified to make sure that the hash string is passed throughout the payment process.

You will need to add this line below into your payment page form[No].html within the html form area.

- `<input type="hidden" name="st_sitesecurity" value="$st_sitesecurity" />`

Once all of the above is complete you will need to inform SecureTrading's support department that you would like to use the security hash feature.

Before this can be enabled you must supply the following details to our support department.

- Your merchant site reference
- The form reference that you would like the hash to be enforced upon (or all forms)
- The list of fields that you will be using to create your hash **along with the order of these fields** (this is very important).
- The password that you will be supplying within the hash
- **Important:** Changing the field order or password without informing SecureTrading will result in transactions being rejected.

We recommend that you include as many unique fields when creating your hash as just including the merchant and currency fields will end up creating the same hash even though the amount field may have been compromised.

When this feature has been enabled and setup correctly any payment that is attempted resulting in an incorrect hash being generated, the payment page will be redisplayed and the variable "sterror" will contain the error message.

The minimum recommended list of fields are merchant, currency, amount, orderreference.

Quick start

This section of the guide shows you how to set up your web site to work with the SecureTrading payment network, and how to test the connection. For the moment, don't worry about customisation of the web pages and email messages to suit your company – this will be dealt with in Section 3.

1.5 Download the template files

Go to <http://www.securetrading.com/support/payment-pages.html> and download the template files required to configure your web site. These are contained in the zip file `template.zip`, (under the heading "Template and example files") which, when unzipped, produces the HTML files and text messages that are used in the transaction process. You can customise all of them to suit your company's requirements, but for the moment, the only one you need to change is `orderpage.html`.

1.6 Customise the order page

`orderpage.html` contains the form where your customer will enter personal details – name, delivery address, telephone number etc. Clicking on the **Click here to enter Credit Card details** button at the bottom of this form takes your customer to the SecureTrading secure server to enter his or her credit/debit card details.

You must make two changes to the order page before it will work correctly: you need to include your site reference and the address for the merchant emails. Let's assume that your site reference is `mysitereference1234` and that you want the merchant emails sent to `forename.surname@mywebsite.co.uk`

1. Open `orderpage.html` in a text editor and find the line that reads:

```
<INPUT TYPE=hidden NAME="merchant" VALUE="test">
```

2. Replace the word `test` with your site reference. The line should now read:

```
<INPUT TYPE=hidden NAME="merchant" VALUE="mysitereference1234">
```

3. Now find the line that reads:

```
<INPUT TYPE=hidden NAME="merchantemail" VALUE="me@mysite.co.uk">
```

4. Replace `me@mysite.co.uk` with your merchant email address, so the line now reads:

```
<INPUT TYPE=hidden NAME="merchantemail"  
VALUE="forename.surname@mywebsite.co.uk">
```

Take care not to alter anything else on these lines, and be especially careful that you don't accidentally remove the quotation marks around the site reference and the merchant email address.

You MUST enter your site reference before you do any testing, otherwise the SecureTrading system will not respond when you submit the form. You can disable the merchant emails if you want (see later), but we strongly recommend that you don't – especially while you are testing the system.

One other item of information is required before the SecureTrading system will respond to the form, namely, the value of the transaction. Find the line:

```
<INPUT TYPE=hidden NAME="amount" VALUE="9999">
```

in the source code of the page. This line sets the value of the transaction in **base units** of the default currency – in this case, UK pennies. So when you submit the form, the transaction will be for £99.99, and you should see this amount on the confirmation web pages and emails.

When you have saved your changes, upload the modified page to your merchant web server (**not** the SecureTrading secure server). You can rename the order page if you want: `orderpage.html` is just a suggested filename, and renaming it will not affect correct operation in any way.

1.7 Testing

1.7.1 Successful transaction

You can now test the integration of your site with SecureTrading's payment network. Start by testing whether the system will handle a successful transaction correctly:

1. Browse to the updated `orderpage.html` on your merchant web server.
2. Fill in some details on this form. In particular, include a valid email address so that you can test that your customers will receive a confirmation email (for testing purposes, you can use the merchant email address, if you want).
3. Click on the **Click here to enter Credit Card details** button. You should be taken to the default credit card details page (`form.html`) on the SecureTrading secure server, and you should see the details that you entered on the previous form.
4. Enter the following credit card details:

Credit Card Type:	Visa
Credit Card Number:	4111 1111 1111 1111
If Switch, Issue Number:	Leave blank
Expires End:	05/10 (or any date that has not passed)

5. Now click the **Send Payment** button.

These credit card details should result in a successful transaction, so after a few seconds, you should see the default `success.html` page from the SecureTrading secure server. Also, you should shortly receive two emails:

- One sent to the merchant email address (the address embedded in the `orderpage.html` file)
- One to the customer's email address, as entered on the order page form.

These may arrive in any order, depending on conditions on the internet.

1.7.2 Unsuccessful transaction

If all has worked correctly, then test to see if the system will handle an unsuccessful transaction correctly:

1. As before, browse to `orderpage.html` on your merchant web server and fill in some details. Submit the form.
2. On the credit card details page, enter the following:

Credit Card Type:	Visa
Credit Card Number:	4242 4242 4242 4242

If Switch, Issue Number:	Leave blank
Expires End:	05/10 (or any date that has not passed)

3. Now click the **Send Payment** button.

These credit card details will result in an unsuccessful transaction. You should see the default `failure.html` page from the SecureTrading secure server, and a notification email will be sent to the merchant email address.

1.7.3 Further testing

You can use the credit card details above to perform any further testing that you wish to do. The 4111... number will always result in a successful transaction, and the 4242... number will always result in an unsuccessful transaction. For other valid sets of credit card details, a successful outcome will be returned. However, we do not recommend that you use real credit card numbers for testing purposes.

2 Advanced customisation

This section of the guide explains how to customise the default web pages and email messages that are used by our SecureTrading Payment Pages service. You can add logos, change the wording and change the layout of the files, providing you take care not to delete the special fields and variables that are needed for correct operation.

2.1 The customisable files

If you have not done so already, download the file `template.zip` from <http://www.securetrading.com/support/payment-pages.html> and unzip it. This produces the following files:

<code>orderpage.html</code>	A template order page
<code>form.html</code>	A template payment form
<code>success.html</code>	The web page your customers see if the transaction is successful.
<code>customeremail.txt</code>	The email sent to the customer following a successfully authorised transaction.
<code>merchantemail.txt</code>	The email sent to the merchant following a successfully authorised transaction.
<code>failure.html</code>	The web page your customers see if the transaction is unsuccessful.
<code>failureemail.txt</code>	The email sent to the merchant following an unsuccessful transaction.

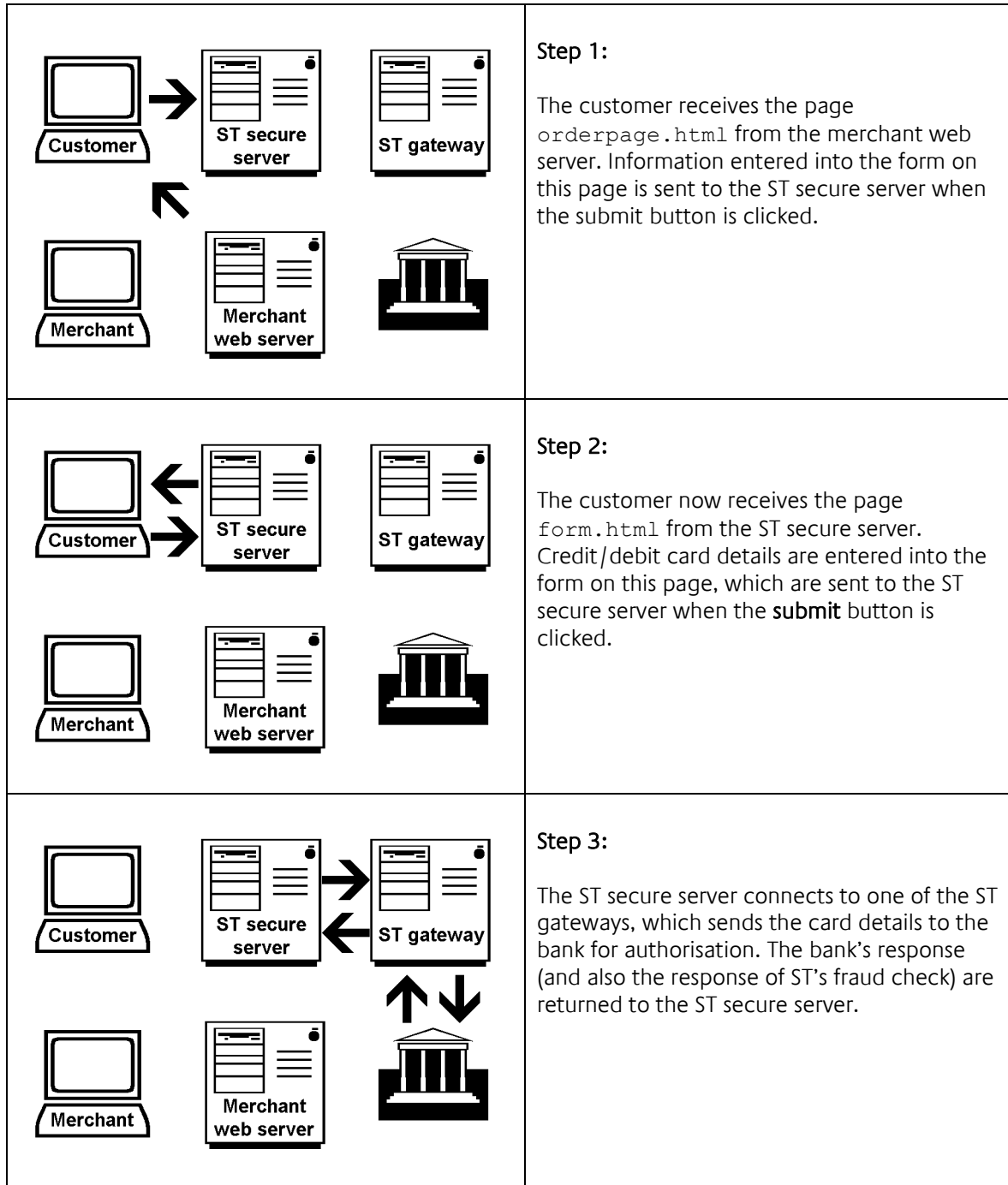
You may also want to download `examplefiles.zip` from the same page, which contains examples of customised versions of the above files.

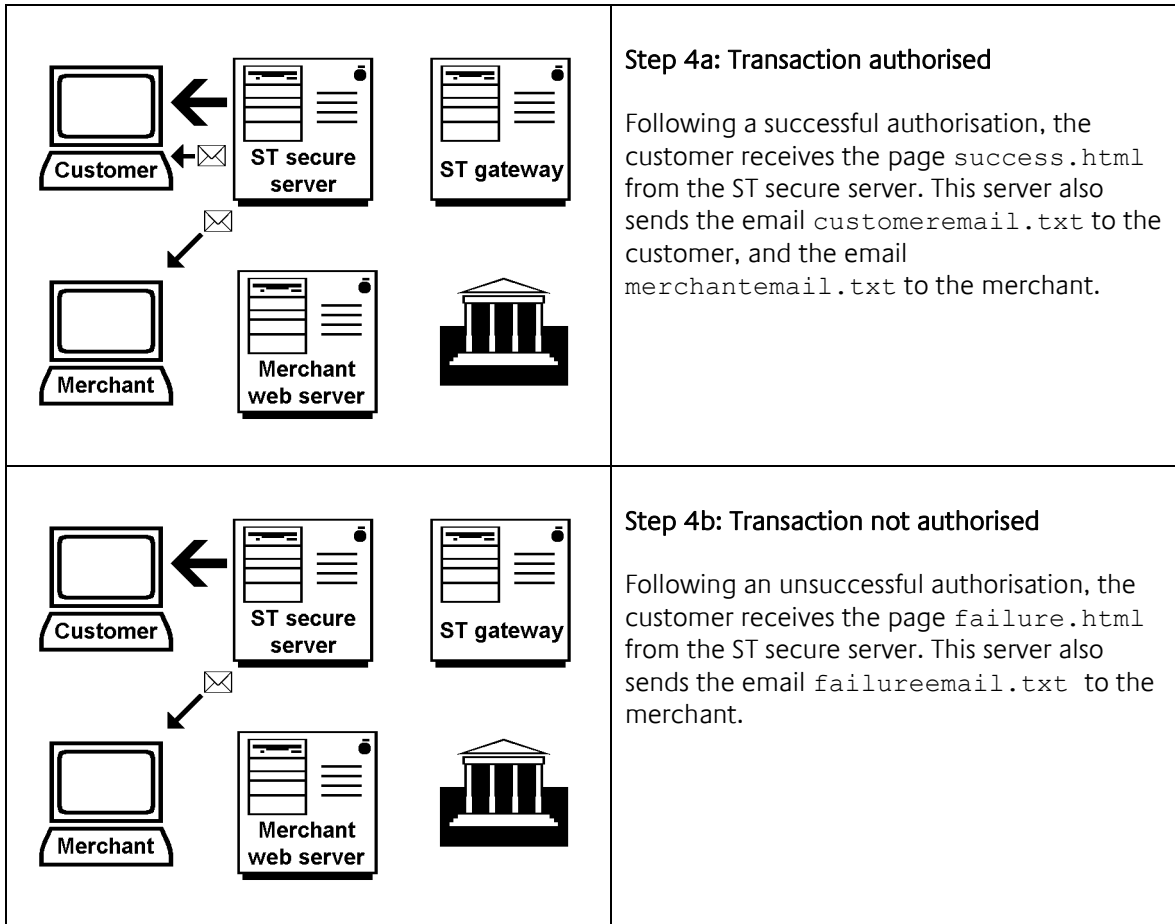
Of these files, `orderpage.html` is stored on your merchant web server; the other files should, after modification, be uploaded to the SecureTrading secure server using the **My-ST** utility (see the **My-ST Guide** for more details – you can download it from <http://www.securetrading.com/support/general-setup-guides.html>)

2.2 The transaction process

The table below shows you how a transaction is processed, and how each of the customisable files is used. Data is passed between the servers using the HTTP POST method, and back to the user by means of html file templates and email templates that contain variables.

Note that steps **4a** and **4b** may be different if you are using our **callback** feature. See the **Callback Guide** for details





2.3 Fields and variables

2.3.1 Fields

The forms in `orderpage.html` and `form.html` contain a number of **fields**, which hold individual items of information that are used in the transaction process. Some of these fields contain data that your customer has entered, for example the customer's name or telephone number. Others – usually hidden from your customer – contain special control information that may affect the way the transaction is handled, such as the currency units (pounds, dollars, etc.), or whether or not your customer receives a confirmation email.

In the HTML source code of a form, a field will look something like this:

```
<INPUT TYPE=text NAME="telephone">
```

This produces a text-entry box where your customer can enter his or her telephone number. The field has the **field name** `telephone`. Field names are important, as they are used to indicate where the value of the field (in this case, the customer's telephone number) will appear in other web pages, and in email messages.

Another example:

```
<INPUT TYPE=hidden NAME="currency" VALUE="gbp">
```

This is a hidden field – your customer does not see it on the form displayed in their web browser. However, it's still a field which is passed to SecureTrading when the **submit** button is clicked. As this field is hidden, the value of the field (in this case, `gbp`) needs to be specified within the HTML code.

Note: Be careful not to remove the quotation marks around the field name and field value when you are editing fields. The system may behave unpredictably if they are missing.

2.3.2 Essential fields

Fields can be added or removed from `orderpage.html` and `form.html` to suit your site. However, as was explained in the **Quick Start** section, two fields are essential: `merchant` and the value of the transaction, in either `amount` or `inputamount`. In the line:

```
<INPUT TYPE=hidden NAME="merchant" VALUE="test">
```

You **must** replace the word `test` with your merchant site reference. You will receive no response from our system if either of these fields is missing.

We also **strongly recommend** that the confirmation email `merchantemail.txt` is sent to you following each transaction, to help you keep track of the orders that you receive. Even if you are using **callback**, the emails are useful as a “back up” if anything goes wrong with the script processing. So, in the line:

```
<INPUT TYPE=hidden NAME="merchantemail" VALUE="forename.surname@mywebsite.co.uk">
```

you should replace the email address `me@mysite.co.uk` with the address to which merchant emails should be sent. If you remove this line completely, no merchant emails will be sent.

2.3.3 Variables

Once the forms containing the fields have been submitted, you can show the values of the fields in subsequent web pages and emails. You do this by inserting **variables** into web page and email templates to act as “placeholders” for the data held in the fields. These variables are replaced by the actual data when the web page or email is sent. Variables are written as the corresponding field name preceded by `$`. For example, `customeremail.txt` contains the following lines:

```
town           : $town
county        : $county
```

When this is sent to the customer, `$town` and `$county` will be replaced by the town and county names that the customer entered in `orderpage.html`, so the customer will receive:

```
town           : Bangor
county        : Gwynedd
```

For comparison, the lines in `orderpage.html` which define the values of these variables are:

```
Please enter delivery Postal Town: <INPUT TYPE=text NAME="town"><BR>
Please enter delivery County/Area: <INPUT TYPE=text NAME="county"><BR>
```

For a full list of fields and variables that the SecureTrading system uses, see the **reference** section at the end of this guide.

2.3.4 User defined fields and variables

You can define your own fields, and pass them to other pages and emails within the system by referencing them as a variable called *\$fieldname*.

For example, you could define a field called `colour` in the form in `orderpage.html`, like this:

```
Enter the item colour: <INPUT TYPE=text NAME="colour">
```

and refer to it as `$colour` in a customer or merchant email.

User-defined field and variable names should consist of **lower case letters** and/or **numbers** only. There are some reserved names that you can't use – see the reference section for more details.

Note that user-defined fields and variables are **not** logged by the SecureTrading secure server.

2.4 Logging

The SecureTrading secure server logs many of the fields and variables listed in the reference section. Because of this, we recommend that you use these field and variables for their intended purpose, and that you don't replace them with your own user-defined fields.

For example, if you use the `address`, `town`, `country` and `postcode` fields to gather your customer's address, like this:

```
Please enter delivery Address: <INPUT TYPE=text NAME="address"><BR>
Please enter delivery Postal Town: <INPUT TYPE=text NAME="town"><BR>
Please enter delivery County/Area: <INPUT TYPE=text NAME="county"><BR>
Please enter delivery Postcode: <INPUT TYPE=text NAME="postcode"><BR>
```

then the address will be logged. If, however, you use your own fields, like this:

```
Please enter delivery Address: <INPUT TYPE=text NAME="address1"><BR>
Please enter delivery Postal Town: <INPUT TYPE=text NAME="address2"><BR>
Please enter delivery County/Area: <INPUT TYPE=text NAME="address3"><BR>
Please enter delivery Postcode: <INPUT TYPE=text NAME="address4"><BR>
```

then the address will **not** be logged, as the fields `address1`, `address2`, `address3` and `address4` are not recognised as fields that need to be logged by the SecureTrading system. And if you used a mixture of user-defined and standard fields, like this:

```
Please enter delivery Address: <INPUT TYPE=text NAME="address1"><BR>
Please enter delivery Postal Town: <INPUT TYPE=text NAME="address2"><BR>
Please enter delivery County/Area: <INPUT TYPE=text NAME="address3"><BR>
Please enter delivery Postcode: <INPUT TYPE=text NAME="postcode"><BR>
```

then only the `postcode` field would be logged.

You may want to define a new field to hold information that relates to your specific product or service, for example, `deliverydate`, and have this field logged on the SecureTrading secure server. If so, then you should use the **Callback** feature, covered in the **Callback Guide**

2.5 Customising individual files

2.5.1 orderpage.html

Strictly speaking, `orderpage.html` as such is not necessary for the system to work. However, you do need a page containing a form, or running a script, which passes information to `form.cgi` on the SecureTrading secure server using the HTTP POST method when the customer clicks on the *submit* button. `form.cgi` is a script which passes the information entered on `orderpage.html` to the correct `form.html` payment page. It is part of the SecureTrading system, and cannot be edited.

For the purposes of this guide we will assume that this page is called `orderpage.html`, and that it uses a HTML form to send the data.

As a bare minimum, the form **must** contain the following lines:

```
<form method="POST" action="https://securetrading.net/authorize/form.cgi">
  <input type=hidden name="merchant" value="site_reference">
  <input type=hidden name="amount" value="transaction_amount">
  <input type=submit value="Click here to order">
</form>
```

where your site reference replaces `site_reference`, and the value of the transaction replaces `transaction_amount`. (note: `inputamount` can be used instead of `amount`.)

You can modify the appearance of `orderpage.html` (or its equivalent) so that it fits in with the style of the other pages on your merchant web site – there are no restrictions at all on doing this. You can even use a different name for this file, if you want – it won't affect the operation of the system.

Our example `orderpage.html` gathers personal details from your customer – name, delivery address, telephone number, etc. – but does **not** ask for card details, as the merchant web server may not be secure.

One field you might want to modify is `requiredfields`. Look for the line:

```
<INPUT TYPE=hidden NAME="requiredfields" VALUE="name,email">
```

in the source code for the page. `requiredfields` holds a comma-separated list of fields that **must** be filled in by your customer: in this case, name and email. If these fields are left blank, then the form is re-displayed to the customer with a red star next to the required fields that they have left blank. Add extra fields to this list as you require. For example, if you change the line so it reads:

```
<INPUT TYPE=hidden NAME="requiredfields"VALUE="name,email,telephone,postcode">
```

then your customer **must** enter something into the telephone and postcode fields on the form.

2.5.2 form.html

This page is displayed to your customer immediately after they submit the form on `orderpage.html`. It is here, on the secure server, that your customer is asked for credit or debit card details.

The default `form.html` displays the information that was sent from `orderpage.html`, and gets the card information from the customer. This information is passed to `process.cgi` on the SecureTrading secure server when the submit button is clicked.

`process.cgi` is the script which sends the data to the payment gateway for authorisation, and sends the appropriate emails and web pages to the customer and merchant, depending on the results of the authorisation. It is part of the SecureTrading system, and cannot be edited.

Only the fields that are sent to `process.cgi` can be referenced by variables in these email messages and web pages. This means that fields that were sent from the form on `orderpage.html` need to be re-entered on the form on this page before the form is POSTed to `process.cgi`. For example, `orderpage.html` asks for the customer's fax number:

```
Please enter Fax Number: <INPUT TYPE=text NAME="fax"><BR>
```

If you want to use your customer's fax number in `customeremail.txt`, for example, then you need to put a field in `form.html` to hold the customer's fax number. Your customer does not have to re-enter their fax number – you can copy the information that was sent from `orderpage.html` using a hidden field and a variable in `form.html`, like this:

```
<INPUT TYPE=hidden NAME="fax" VALUE="$fax">
```

The `fax` field will now be sent to `process.cgi` when the `submit` button is clicked. You need to do the same for all the fields that will be processed, or are used, in later web pages and emails. Be especially careful to pass along the `merchant` and `amount` fields – `form.html` **must** contain the following two lines:

```
<INPUT TYPE=hidden NAME="merchant" VALUE="$merchant">
<INPUT TYPE=hidden NAME="amount" VALUE="$amount">
```

(Note: You can use the `inputamount` field instead of `amount`, if you want. See the reference section of this guide for the difference between the two.)

You can customise this page to your own requirements, **provided the changes do not breach Article 4.1 of the SecureTrading Merchant Agreement.**

2.5.3 success.html

This page is displayed to your customer after a successful transaction. By default, `success.html` displays the values of many of the fields sent by `form.html`, and also shows some fields that you may find useful for debugging your web pages (these should be removed from the page before you go "live").

You can customise this page to your own requirements, **provided the changes do not breach Article 4.1 of the SecureTrading Merchant Agreement.**

Note: This page can be suppressed if you use `callback` – see the **Callback Guide** for details.

2.5.4 failure.html

This page is displayed to your customer after an unsuccessful transaction. By default, it contains the same information as the success page, including the debugging information.

You can customise this page to your own requirements, **provided the changes do not breach Article 4.1 of the SecureTrading Merchant Agreement.**

Note: This page can be suppressed if you use `callback` – see the **Callback Guide** for details.

2.5.5 merchantemail.txt

This file contains the text of the email that is sent to the merchant after a successful transaction. By default, this simply lists the information entered from the default `orderpage.html` and `form.html` pages, and other fields generated by the SecureTrading secure server and the bank server. You can modify the text of the email, and add or remove variables, to suit your needs.

Be careful when editing the header of the email, ie the section at the top that reads:

```
To: $merchantemail
From: "$name" <confirmation@securetrading.com>
Subject: Credit Card Order
Reply-To: "$name" <$email>
```

You can change the subject line, but leave the other lines alone, or else you may stop the email from being sent. You should also ensure there is a blank line between the final header and the start of your email text.

2.5.6 customeremail.txt

This is the email that is sent to your customer after a successful transaction. As with `merchantemail.txt`, you can customise the email to your own requirements, but be careful when editing the header:

```
To: "$name" <$email>
From: "securetrading" <confirmation@securetrading.com>
Subject: Credit Card Order
Reply-To: "securetrading" <$merchantemail>
```

You can change the subject line, and you may wish to change “securetrading” to your own company name (the “quotes” are still required) but leave the other lines alone, or else you may stop the email from being sent. You should also ensure there is a blank line between the final header and the start of your email text.

The customer email does not have to be sent. Whether it is sent or not is governed by the `customeremail` field in `orderpage.html` or `form.html`. In `orderpage.html` look for the line that reads:

```
<INPUT TYPE=hidden NAME="customeremail" VALUE="1">
```

If the value of `customeremail` is set to 1, then the customer receives a confirmation email. Change the 1 to 0 if you don't want the customer to receive an email.

2.5.7 failureemail.txt

This is the email that is sent to the merchant after an unsuccessful transaction. Once again, you can customise this email to your own needs, but be careful when editing the header (see `merchantemail.txt` above).

Note: `failureemail.txt` is sent to the merchant only if the variable `$merchantemail` contains a valid email address for the merchant. Make sure you have modified `orderpage.html` so that it contains the merchant email address, as described in the **quick start** section of this guide.

2.5.8 UTF-8 email character encoding

Customer, Merchant and Failure emails are sent without character encoding headers by default. If you would like to send these emails using the UTF-8 character set headers, you will need to complete the following steps.

1. When the Customer is redirected to your form page on SecureTrading's webserver, the Customer's browser must have been set to UTF-8 encoding. This can be done, for example, by setting the META data on the individual pages or by configuring the page headers on your webserver. This should set the character set used by the Customer's browser to UTF-8, and any form data provided by the Customer should then be encoded in UTF-8.
2. You will also need to pass through the hidden field "st_emailencoding". The value of this field should be set to "utf-8". If this field is present with the correct value SecureTrading will send the email with a "charset" of "utf-8" in the Content-Type header.

```
<INPUT TYPE=hidden NAME="st_emailencoding" VALUE="utf-8">
```

3. If you have customised any of the merchantemail.txt, failureemail.txt or customeremail.txt files to include any non-ASCII characters, you will need to encode these files in UTF-8 and then upload them with the file manager. This can be done using a UTF-8 enabled text editor.

Note: If the value of the st_emailencoding field is not present, or is not set to "utf-8", SecureTrading will default to sending emails with no email encoding header.

2.6 Images in customised web pages

You may want to display an image, for example a .jpg or .gif file of your company logo, on one of your web pages held on the SecureTrading secure server. If so, then the image that you are using should also be located on the SecureTrading secure server, so that the entire content of the particular page that your customer sees comes from a secure server, not from a mixture of secure and insecure servers. This means that your customer will not experience annoying (and to the inexperienced, worrying) dialogue boxes warning about content being downloaded from insecure servers.

Let's say that you want to insert a logo into form.html (or success.html or failure.html) which is called companylogo.gif.

You can insert the logo into the web page using HTML code like this:

```
<IMG SRC="$path/companylogo.gif">
```

The \$path variable holds the full path to your files on the SecureTrading secure server. Its value is generated automatically by the SecureTrading system – you should **not** enter it as a field in any web page.

Add all other images in the same way. You need to upload the image files to the SecureTrading secure server before they can be used on your pages. Use the upload manager of **My-ST** to do this.

2.7 Multiple payment pages

You can have more than one payment page, if you want. Copy the template files to a new set of files with a number appended to the end of the filename, for example:

```
form.html           -> form2.html
success.html       -> success2.html
failure.html       -> failure2.html
customeremail.txt  -> customeremail2.txt
failureemail.txt   -> failureemail2.txt
merchantemail.txt  -> merchantemail2.txt
```

Use 3, 4, 5... instead of 2 for further sets of files, as needed. Customise the files as necessary, and upload them to the SecureTrading secure server (instructions are in the **My-ST Guide**).

To use this set of files instead of any other set, you need to add a `formref` field to the order page (`orderpage2.html`, `orderpage3.html`, or whatever you have decided to call them) and to the payment page (`form2.html`, `form3.html`, etc.).

Order page: Inside the order form that is sent to SecureTrading when the submit button is clicked, add the following line:

```
<INPUT TYPE="HIDDEN" NAME="formref" VALUE="2">
```

This will cause the SecureTrading system to use the files whose names end in 2 (change the 2 to 3, 4, 5... to use other sets of files).

Payment page: At the start of the form inside the payment page are a number of hidden fields. Add the following line to the end of the list:

```
<INPUT TYPE="HIDDEN" NAME="formref" VALUE="$formref">
```

After you have uploaded the modified files, you should be able to use multiple payment pages.

2.8 Uploading files

The following customised files **must** be uploaded to the SecureTrading secure server:

```
customeremail.txt  
failure.html  
failureemail.txt  
form.html  
merchantemail.txt  
success.html
```

You must **not** change the names of these files, otherwise the default version of the file (ie. the original template file) will be used instead. Be careful not to alter the case of the names – the file names must be entirely in lower case.

You also need to upload any image files that are used on `failure.html`, `form.html` or `success.html`, and any extra sets of payment pages that you have created.

You can upload these files to the SecureTrading secure server by using the upload manager of **My-ST** – see the **My-ST Guide** for details of how to do this.

3 Example using custom fields

In this example, we will show you how to modify the original template files so that they handle an extra item of data. Let's say that you wanted to get an alternative phone number, such as a mobile phone number, from your customers, and you wanted to display this number on your success page and confirmation emails. Here's how you would go about it:

Step 1: Modify orderpage.html

You need to insert a field in the form on this page so that your customer can enter the mobile phone number. Open `orderpage.html` and look for the lines:

```
Please enter delivery Country: <INPUT TYPE=text NAME="country"><BR>
Please enter delivery Post Code/ZIP Code: <INPUT TYPE=text NAME="postcode"><BR>
Please enter Telephone Number: <INPUT TYPE=text NAME="telephone"><BR>
Please enter Fax Number: <INPUT TYPE=text NAME="fax"><BR>
```

Between the "Telephone Number" and "Fax Number" lines, insert a new line to get the mobile number from your customer and store it in a field called `mobile`. Your code should now look like this:

```
Please enter delivery Country: <INPUT TYPE=text NAME="country"><BR>
Please enter delivery Post Code/ZIP Code: <INPUT TYPE=text NAME="postcode"><BR>
Please enter Telephone Number: <INPUT TYPE=text NAME="telephone"><BR>
Please enter Mobile Number: <INPUT TYPE=text NAME="mobile"><BR>
Please enter Fax Number: <INPUT TYPE=text NAME="fax"><BR>
```

Save the file and upload it to your merchant web server.

Step 2: Modify form.html

The mobile number needs to be passed from `form.html` to `process.cgi` if you want to use it in the success page and the confirmation emails. So, it needs to appear as a field within the form on this page. Assume for now that you don't want to display the mobile number on this page. Open `form.html` and look for the lines:

```
<INPUT TYPE=hidden NAME="callbackurl" VALUE="$callbackurl">
<INPUT TYPE=hidden NAME="failureurl" VALUE="$failureurl">
<INPUT TYPE=hidden NAME="fax" VALUE="$fax">
<INPUT TYPE=hidden NAME="url" VALUE="$url">
```

After the "url" line, add another hidden field to hold the mobile number, so your code now looks like this:

```
<INPUT TYPE=hidden NAME="callbackurl" VALUE="$callbackurl">
<INPUT TYPE=hidden NAME="failureurl" VALUE="$failureurl">
<INPUT TYPE=hidden NAME="fax" VALUE="$fax">
<INPUT TYPE=hidden NAME="url" VALUE="$url">
<INPUT TYPE=hidden NAME="mobile" VALUE="$mobile">
```

When the page is sent to your customer, the variable `$mobile` will be replaced with the mobile number that he or she entered, so the number will be passed on when the form is submitted. Save the page, and upload it to the SecureTrading secure server.

Step 3: Modify success.html

You need to insert the variable `$mobile` on this page to display the mobile number. Open `success.html` and look for the lines:

```
postcode      : $postcode
country       : $country
telephone     : $telephone
fax           : $fax
```

Between the “telephone” and “fax” lines insert a new line to display the mobile number, so that the code now looks like this:

```
postcode      : $postcode
country       : $country
telephone     : $telephone
mobile       : $mobile
fax           : $fax
```

`$mobile` is replaced by the mobile number that your customer entered when the page is sent from the web server. Once again, save the page, and upload it to the SecureTrading secure server.

Step 4: modify the confirmation emails

Both these emails need the same modification – namely, you need to insert the variable `$mobile` into each one to show the mobile number. Open these emails in turn and look for the lines:

```
telephone     : $telephone
fax           : $fax
```

Add a line between these two lines to display the mobile number, so the code now reads:

```
telephone     : $telephone
mobile       : $mobile
fax           : $fax
```

Save the files and upload them to the SecureTrading secure server.

Step 5: Testing

Browse to the modified `orderpage.html` on your merchant web server and send a test transaction, using the credit card details given in the **quick start** section. You should see the mobile number you entered on the success page and on both confirmation emails.

4 Reference

4.1 Files

File name	Location		Description
	Merchant web server	ST secure server	
orderpage.html (note 1)	*		The page on the merchant web server containing the form which calls <code>form.cgi</code> on the SecureTrading secure server when submitted.
form.html		*	The page on the SecureTrading secure server containing the form where customers enter credit card details. This form calls <code>process.cgi</code> when submitted.
success.html		*	The page displayed to customer after a successful transaction (if <code>callbackurl\$<>1</code> , <code>pipe=no</code> (note 3))
failure.html		*	The page displayed to the customer after declined transaction (if <code>failureurl\$<>1</code> , <code>pipe=no</code> (note 3))
customeremail.txt		*	The email sent to the customer after a successful transaction (if <code>customeremail\$=1</code>)
merchantemail.txt		*	The email sent to the merchant after a successful transaction
failureemail.txt		*	The email sent to the merchant after a declined transaction
callback.txt		*	Called after a successful transaction if <code>callbackurl\$=1</code> (note 2)
callback-f.txt		*	Called after a declined transaction if <code>failureurl=1</code> (note 2)
testcallback.cgi (note 3)	*		Script called during transaction

Note 1: Suggested filename only.

Note 2: Optional **callback** feature, ignore completely for non-callback users.

Note 3: Optional **callback** feature, ignore "if..." part for non-callback users.

Files on the SecureTrading secure server **must** have above filenames, and **must** be in lower case.

Default and example customised files are available from:

<http://www.securetrading.com/support/payment-pages.html> in the TEMPLATES AND EXAMPLE FILES section.

4.2 Fields and variables

Field name	Notes	Description
address		The first line of the customer's address
amount	2	Transaction amount in base units (pence, cents, etc.). For example, £ 10.00 is 1000 in base units
ccissue		Issue number of card (only required for Switch cards. Leave blank for credit cards)
ccnumber	4	The credit card number input by the customer
cctype		Credit card type (Mastercard, Visa, Amex, Delta)
comments		Comments from the customer
company		The customer's company
country		The customer's country
county		The customer's county
currdate		Current date, eg. 31/12/2004
currdays		Current day, eg. Friday
currency	5	The transaction currency (defaults to gbp - UK £ sterling)
currtime		Current time, eg. 10:02:01
email	3	The customer's email address
fax		The customer's fax number
formattedamount	6	A formatted version of the field amount incorporating the appropriate currency sign, eg. £ 10 is £10.00)
formatteddate		Formatted date, eg. Fri Dec 31, 2004
inputamount	2	Transaction amount in main units (pounds, dollars, etc.). For example, £ 10 is 10.00
merchant	1	Your site reference
month		Expiry month (eg. 04)
name	3	The customer's name
orderref		Your reference for this transaction
orderinfo		Your order information
path	6	The absolute path to your ST SECURE SERVER directory. If you want your logo shown within form.html you must use <code></code>
postcode		The customer's postcode
random8	6	An 8 digit random number
random16	6	A 16 digit random number
random32	6	A 32 digit random number
settlementday		Number of days to delay settlement. 0=never settle, 1=settle in the next batch (default), 2=settle in tomorrow's batch, etc.
securitymessage	8	Text description of the response of security code checking
securityresponseaddress	8	Numeric representation of security code checking (address)
securityresponsepostcode	8	Numeric representation of security code checking (postcode)
securityresponsesecuritycode	8	Numeric representation of security code checking (security code)
startmonth		Start month on the card (if applicable)
startyear		Start year on the card (if applicable)
stauthcode	7	The authorisation code received from the bank

Field name	Notes	Description
stconfidence	7	The confidence level returned by the SecureTrading Fraud System. The results range from +100 to -100 (A value of 0 means no information (most common))
streference	7	The ST Reference Number for this transaction
stresult	7	The ST result code 1=pass; 2=fail
st_emailencoding		See UTF-8 email character encoding section
telephone		The customer's telephone number
timestamp		Timestamp, eg. 9218 52352 (changes per second)
town		The customer's town
trunccnumber	6	The truncated (last 4 digits) of the credit card number used
url		The customer's website url
usadate		Current date in USA format, eg. 12/31/2004
year		Expiry year (eg. 2004)

Notes:

1. Mandatory field – **must** be present
2. Mandatory, but use **either** amount **or** inputamount - not both
3. Suggested required fields as governed by requiredfields
4. Value must remain on a secure server, so is not available to merchant
5. The transaction currency can only be one that we support
6. Field generated by the SecureTrading secure server
7. Field generated by the bank server, or extra fields for use in success.html
8. The information in these fields will only be available in the merchant email and in callback. The information should NOT be made available to the customer. For the various responses that may be included in these fields please refer to Appendix 1.

4.3 Control fields

These fields control what happens when a transaction is processed within our system.

Field name	Description
<code>callbackurl</code>	The reference number of the cgi script that will be called with the transaction result in the event of a successful authorisation. The information relating to these script(s) is contained in <code>callback.txt</code> . eg. <code>callbackurl="1"</code>
<code>customeremail</code>	If set to 1 will email confirmation to the customer (if email field not blank). eg. <code>customeremail="1"</code>
<code>failureurl</code>	The reference number of the cgi script that will be called with the transaction result in the event of a declined authorisation. The information relating to these script(s) is contained in <code>callback-f.txt</code> . eg. <code>failureurl="1"</code>
<code>formref</code>	If defined, gives a complete set of new <code>form.html</code> , <code>success.html</code> , <code>email</code> etc. files that are used. If <code>formref</code> is not defined, the default <code>form.html</code> is used. For example if <code>formref</code> is 3, the system will use <code>form3.html</code> , <code>success3.html</code> , <code>customeremail3.txt</code> , etc. eg. <code>formref="1"</code>
<code>merchantemail</code>	If it contains a valid email address, confirmation will be emailed to the merchant at that address. eg. <code>merchantemail="forename.surname@mywebsite.co.uk"</code>
<code>requiredfields</code>	A comma-separated list of required fields to be filled in on the payment request form. If any of these fields are not filled in by the customer, the form will be re-presented with a red star by each omitted field. eg. <code>requiredfields="name,telephone,email"</code>

4.4 User defined fields

You can define other fields, and pass them to other pages and emails within the system by referencing them as a variable called `$fieldname`.

For example, you could define a field called **colour** in the form in `orderpage.html`, like this:

```
<INPUT TYPE=text NAME="colour">
```

and refer to it as `$colour` in a customer or merchant email.

User-defined field names should consist of **lower case letters** and/or **numbers** only.

User defined fields are not recorded in the transaction file. They can however be passed back in any emails that are returned, or if you are using the **callback** facility.

4.5 Reserved field names

The following fields are used within the SecureTrading's system and must not be used by the merchant:

Field name	Purpose
<code>language</code>	Language
<code>ipinfo</code>	Browser IP address
<code>browserinfo</code>	Browser Type
<code>referrer</code>	Referred URL
<code>st_???</code>	Any fieldname beginning with 'st_' is a reserved field

4.6 Mandatory fields

Two fields MUST be passed to the SecureTrading secure server for any transaction to take place:

- `merchant` – your site reference
- either `amount` or `inputamount` – the amount to be processed. `amount` is in **base units** (eg. . pence or cents), while `inputamount` is in **main units** (eg. pounds or dollars). Use one or the other of these fields, **but not both**.

4.7 Start/expiry date fields

The form page is used to take the customer's credit card details. This page should have selection boxes for the start date (month and year) and expiry date (month and year) of the credit card.

The selectable year will change over time, so we have provided two fields `$st_startyears` and `$st_expiryyears`. These will insert a drop-down selection box for the start year and expiry year, the box will already contain the relevant year options and will be updated automatically by the system every year.

For an example of using these fields, please refer to the `form.html` page in the `templatefiles.zip` available on our website.

The generated HTML selection boxes have a class set to `st_startyear` and `st_expiryyear`. You can use this class with Style Sheets to customise the style of these boxes. For example, to make the background colour of the start year box yellow put the following code in the head of the page:

```
<STYLE TYPE="text/css">
<!--
.st_startyear    {background:yellow}
-->
</STYLE>
```

4.8 Country codes

It is preferable to specify a country using a 2 character iso code instead of a free text field. Some acquirers require this to process a transaction.

A field `$st_countrycodes` is available to insert a drop-down selection box containing all the countries supported by the system. This may be used in place of the `$country` variable in your forms.

5 Redundancy planning

As with all services, occasionally situations may arise where, due to circumstances beyond SecureTrading's control, the <https://securetrading.net> Payment Pages may be unavailable. As a result, we have developed a back up service which can be switched to by making some simple changes and having them in place. Please remember that, as a merchant, the onus will be on you to ensure that this is all in place to ensure uninterrupted payments in the event of our primary site being unavailable.

The relevant information is set out below, however, please contact Support if you require any further information about this service.

We have created 2 alternative URLs:

<https://pay2.securetrading.net/authorize/form.cgi>
<https://pay3.securetrading.net/authorize/form.cgi>

All payment system functionality that is available via the default site (<https://securetrading.net>) will be available through the alternative sites.

In addition My-ST will be available but certain functionality will NOT be available at the alternative sites:

- * File manager
- * Password changing

Passwords can be changed and payment page template files can be uploaded via <https://securetrading.net/merchantServices> only.

Any file uploaded or passwords changed may take up to two hours to become available via pay2.securetrading.net and pay3.securetrading.net.

NB: HTML pages used by securetrading.net, pay2.securetrading.net and pay3.securetrading.net will operate as normal if all links are converted to relative, i.e. the domain portion must be removed. For example:

"<https://securetrading.net/authorize/process.cgi>"

...must be changed to:

["/authorize/process.cgi"](/authorize/process.cgi)

Please ensure that **all** links in **all** HTML pages have been altered. Further information on relative URLs is available from the W3 website:

<http://www.w3.org/TR/WD-html40-970708/htmlweb.html#h-4.1.2>

When making any changes please ensure that all existing functionality operates as expected:

- * merchant emails (success and failure)
- * customer emails
- * callback (success and failure)

If you have any questions or queries, please don't hesitate to contact the Support team.

6 Appendix 1

The following section details the values returned by the `sesecurityresponsesecuritycode`, `securityresponseaddress`, `securityresponsepostcode` fields as well as their meanings.

Security Code Security Response

Response Value	Response Text
0	NO INFORMATION AVAILABLE
1	DATA NOT CHECKED
2	DATA MATCHED
4	DATA NOT MATCHED

Security Code Address and Security Code PostCode

Response Value	Response Text
0	NO INFORMATION AVAILABLE
1	DATA NOT CHECKED
2	DATA MATCHED
4	DATA NOT MATCHED
8	PARTIAL MATCH

Description of Security Responses

Response Text	Description
DATA NOT CHECKED	The data submitted to the acquiring bank was not checked. Either the details were not passed or the acquiring bank couldn't perform the check based on the card details.
DATA MATCHED	The information sent to the acquiring bank was valid and has passed their security checks
DATA NOT MATCHED	The information sent to the acquiring bank was invalid and has failed their security checks
PARTIAL MATCH	Part of the information sent to the acquiring bank was valid while the remainder was either not checked or invalid.

The following table details the values returned in the `securitymessage` field

Response Message	Description
DATA MATCHED	The information sent to the acquiring bank was valid and the transaction has passed their security checks
SEC CODE MATCH	The security code information sent to the acquiring bank was valid but the address information submitted was invalid, missing or not checked.
ADDRESS MATCH ONLY	The address information sent to the acquiring bank passed the validation but the security code was invalid, missing or not checked.
DATA NOT MATCHED	Both the security code and the address information sent to the acquiring bank were invalid.
DATA NOT CHECKED	The data submitted to the acquiring bank was not checked. Either the details were not passed or the acquiring bank couldn't perform the check based on the card details.