**GP Connect Demonstrator Configuration**

The purpose of this document is to give more detail on how to configure the GP Connect Demonstrator, for example how to configure the demonstrator to work with the Spine Proxy or how to configure the demonstrator to get data from a second suppliers system.

**1. Spine Configuration, Federated Practices & Default Practice**

The GP Connect Demonstrator is configurable to work with the Spine Proxy. The demonstrator can also be configured to work with additional provider systems either acting as federated practices or as the main default supplier system.

The Spine proxy and default practice information are set within two configuration files:

|  |  |
| --- | --- |
| **Location** | **Context Environment Variable Name** |
| \gpconnect-demonstrator-api\ providerRouting.json | gp.connect.provider.routing.file |
| \gpconnect-demonstrator-api\ defaultPracticeOdsCode.html | n/a |

Within the “**providerRouting.json**” file the Spine Proxy URL, federated practice details and default practice ASID is set. The endpoint details and ASID’s for the supplier systems are used if the demonstrator cannot reach the configured SDS server to get the endpoint details. The content of the “providerRouting.json” file is shown below:

*{*

*"spineProxy": "http://spineproxy.nhs.uk/",*

*"ASID": "200000000359",*

*"practices": [*

*{*

*"id": "1",*

*"odsCode": "GPC001",*

*"name": "GP Connect Demonstrator",*

*"interactionIds": ["\*"],*

*"endpointURL": "/fhir",*

*"apiEndpointURL": "/api",*

*"ASID": "200000000360"*

*},*

*{*

*"id": "2",*

*"odsCode": "R1A14",*

*"name": "Test GP Care Trust",*

*"interactionIds": ["\*"],*

*"endpointURL": "http://localhost:18080/fhir",*

*"apiEndpointURL": "http://localhost:18080/api",*

*"ASID": "200000000361"*

*}*

*]*

*}*

|  |  |
| --- | --- |
| spineProxy | This configuration element is where the spine proxy URL should be entered. In the file the default value is “”, an empty string which will cause the demonstrator front end to talk directly to the suppliers fhir endpoint without going through the spine proxy. |
| ASID | This value is the default ASID of the demonstrator, the client ASID which is used when making requests to a suppliers fhir endpoints. |
| Practice details | The object highlighted blue is the configuration for a practice. This could be the default practice or a federated practice. The example file content above shows two practices, one with the ODS code “GPC001” and one with the ODS code “R1A14”. The “endpointURL” element is the supplier systems endpoint for that practice. The “ASID” element is the ASID of the suppliers system, which will be used in the Ssp-To header element when the demonstrator make a request to the fhir endpoint.  The “id”, “name”, “apiEndpointURL” and “interactionIds” elements are only used when the practice is set to be the default practice. They indicate to the demonstrator when the demonstrator supporting api’s are located and some basic information. These elements should always point towards demonstrator server. For the GPC001 practice the URL are relative and point back to the current GP Connect server instance. |

To set the default practice that the demonstrator will use, the corresponding ODS code needs to be set in the “**defaultPracticeOdsCode.html**” file. The contents of the file is very simple, it is a short piece of html which the demonstrator includes on the index.html page so the javascript can pick up the value and use the information to find the default practice from the providerRouting.json file. The content of the file is shown below:

*<div id="defaultPracticeOdsCode">GPC001</div>*

To change the default practice, replace the highlighted “GPC001” value within the file with the ODS code of the practice from within the “providerRouting.json” file. This will change the ODS code the demonstrator uses when performing endpoint lookups and calling the fhir endpoints.

A new addition to the demonstrator was the testing configuration tools as shown in the screen shot below (Figure 1). The testing configuration tools enable a user to set the default practice ODS Code temporarily without changing the configuration file on the server. More specific details on testing tools and their use is shown in “Section ZZZ – Testing Configuration Options”.

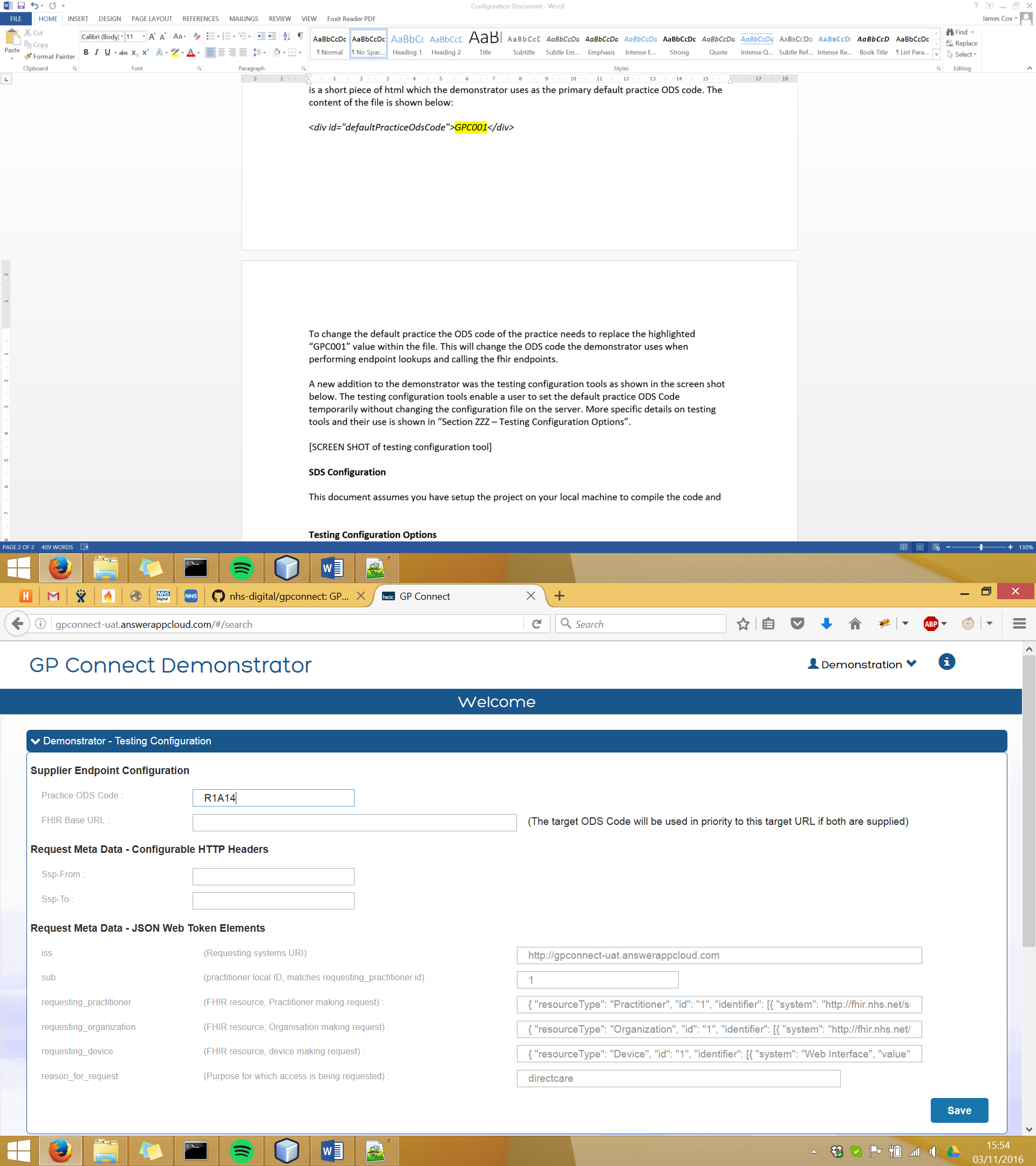


Figure 1 – Demonstrator testing configuration options

**2. Spine Proxy SSL Certificate Configuration**

The SSL certificate configuration for the GP Connect Demonstrator utilises the tomcat server “Connector” configuration for creating an SSL connector on the server and the client SSL and certificate utilises the standard browser certificate and SSL capability.

**Server side certificate configuration**

To configure the SSL port and certificates on the deployed tomcat server you will need to change the “[tomcat\_base]/conf/server.xml” file to contain a connector similar to the one below.

<Connector port="18443"

protocol="org.apache.coyote.http11.Http11NioProtocol"

maxThreads="150"

scheme="https"

secure="true"

SSLEnabled="true"

truststoreFile="C:\GPConnectTomcat\gpconnectServer2\certs\tomcat.keystore"

truststorePass="password"

keystoreFile="C:\GPConnectTomcat\gpconnectServer2\certs\tomcat.keystore"

keystorePass="password"

clientAuth="true"

keyAlias="gpconnect"

sslEnabledProtocols="TLSv1,TLSv1.1,TLSv1.2"

ciphers="TLS\_ECDHE\_RSA\_WITH\_AES\_256\_CBC\_SHA384,TLS\_ECDHE\_RSA\_WITH\_AES\_256\_CBC\_SHA,TLS\_ECDHE\_RSA\_WITH\_RC4\_128\_SHA,TLS\_RSA\_WITH\_AES\_256\_CBC\_SHA256,TLS\_RSA\_WITH\_AES\_256\_CBC\_SHA,SSL\_RSA\_WITH\_RC4\_128\_SHA"

/>

Details of all possible configuration attributes for a connector can be found in the tomcat documentation relating to the tomcat version you are using. The table below describes the important specifics of the connector above.

|  |  |
| --- | --- |
| Attribute | Details |
| protocol | This specifies the library and connection method to use for the connector. In this case it will use the JSSE library to supply the implementation for the connector and the connection will be a non-blocking connection so multiple connections can be made and authenticated at the same time. |
| SSLEnabled | This tells the server to use SSL for this port. |
| truststoreFile | This attribute has to be the absolute path to the servers trust store which contains the client certificates that the server will allow to connect to the server. This is usually a JKS (Java Key Store). |
| truststorePass | This is the password required to access the trust store. |
| keystoreFile | The keystore is the absolute path to the JKS (Java Key Store) that contains the public key for the servers. This is the certificate that the server will sent to the client during the SSL handshake. |
| keystorePass | This is the password to access the keystore file. |
| clientAuth | This attribute is important as it controls the requirement for client authentication. When this attribute is “true” the server will request a certificate from the client during the authentication process. This is required as the GP Connect Demonstrator specifies that the SSL should use mutual authentication. |
| sslEnabledProtocols | In tomcat this is an important attribute to include as by default the tomcat server will only use TLSv1 and TLSv1.1 which will not allow AES-256 ciphers. TLSv1.2 needs to be included in the allowed protocols to allow the use of the AES-256 ciphers. This is a requirement as the Spine team have created a short list of acceptable ciphers and have specified that the encryption should use AES-256. |
| ciphers | This attribute is used to list the acceptable ciphers the server should accept from the client. The list in the example above is the current list at the time of writing this document but may change so please check in the GP Connect specification for the current requirements. |

**Browser Certificate Configuration**

For the required mutual authentication the browser will need to supply the server with a certificate during the SSL handshake. Most browsers are capable of sending a certificate to the server with some simple configuration. Due to differences in browser implementation there is sometimes problems with the browser sending the certificate during the CORS (cross site scripting) OPTIONS if the demonstrator is calling a different suppliers backend or a second instance of the GP Connect demonstrator on another server or port on the same server. The issue is that the OPTIONS call is done over SSL but some browsers do not send the client certificate to the server on the OPTIONS call which make authentication fail and the client never calls the fhir endpoint for the request.

The Google Chrome browser does send the client certificate with the OPTIONS call so does not have the issue. To setup the Chrome to send the client certificate you need to go into the options menu and select “Settings”. At the bottom of the setting window there is an option for advanced settings. If you click the advanced settings option and scroll down you will see a button labelled “Manage certificates…” (Figure 2).

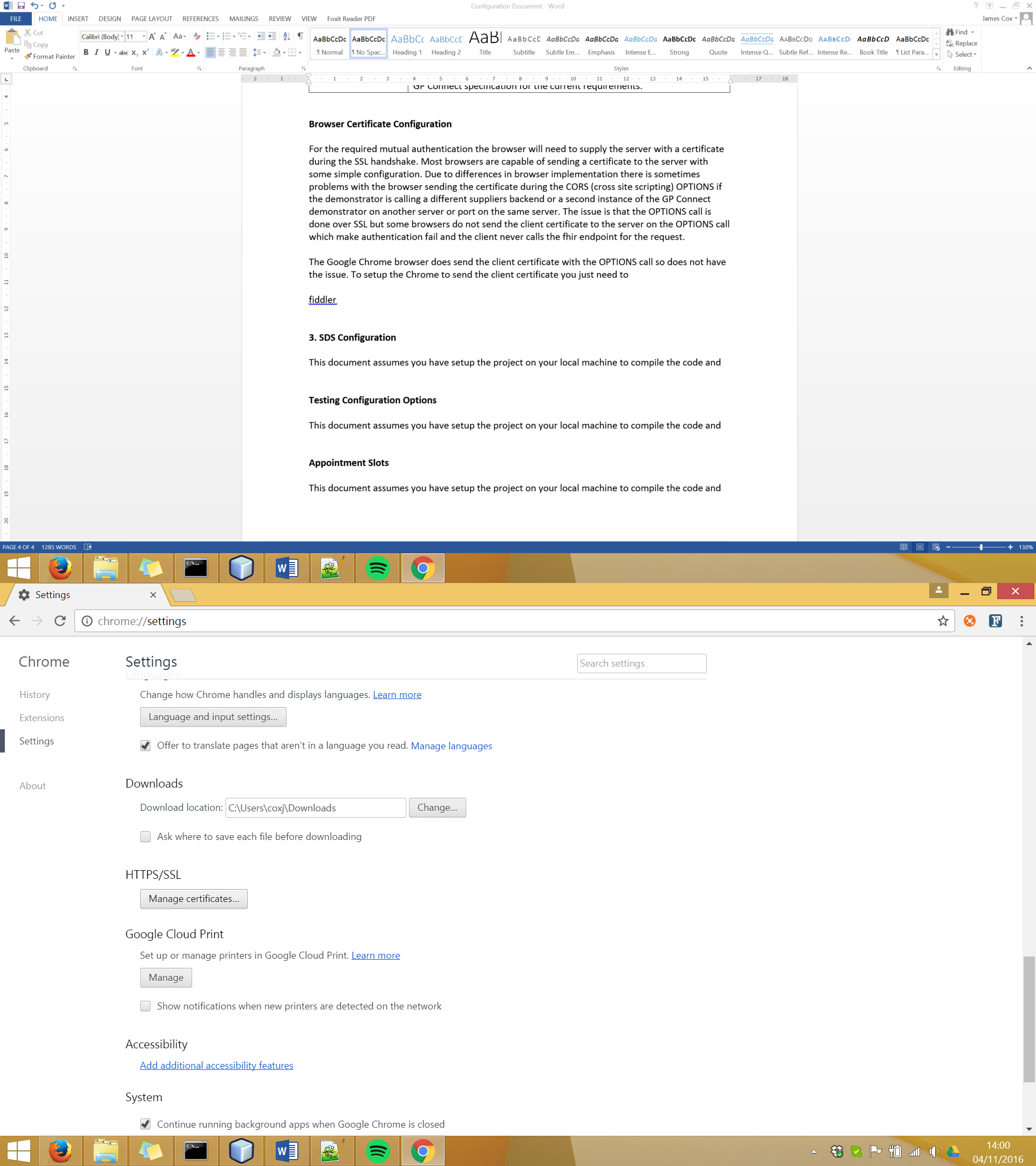


Figure 2 – Google Chrome manage certificates option

If you click the manage certificates button a popup will appear (Figure 3) which will allow you to import a certificate file into Chrome which it can use for mutual authentication with the server.

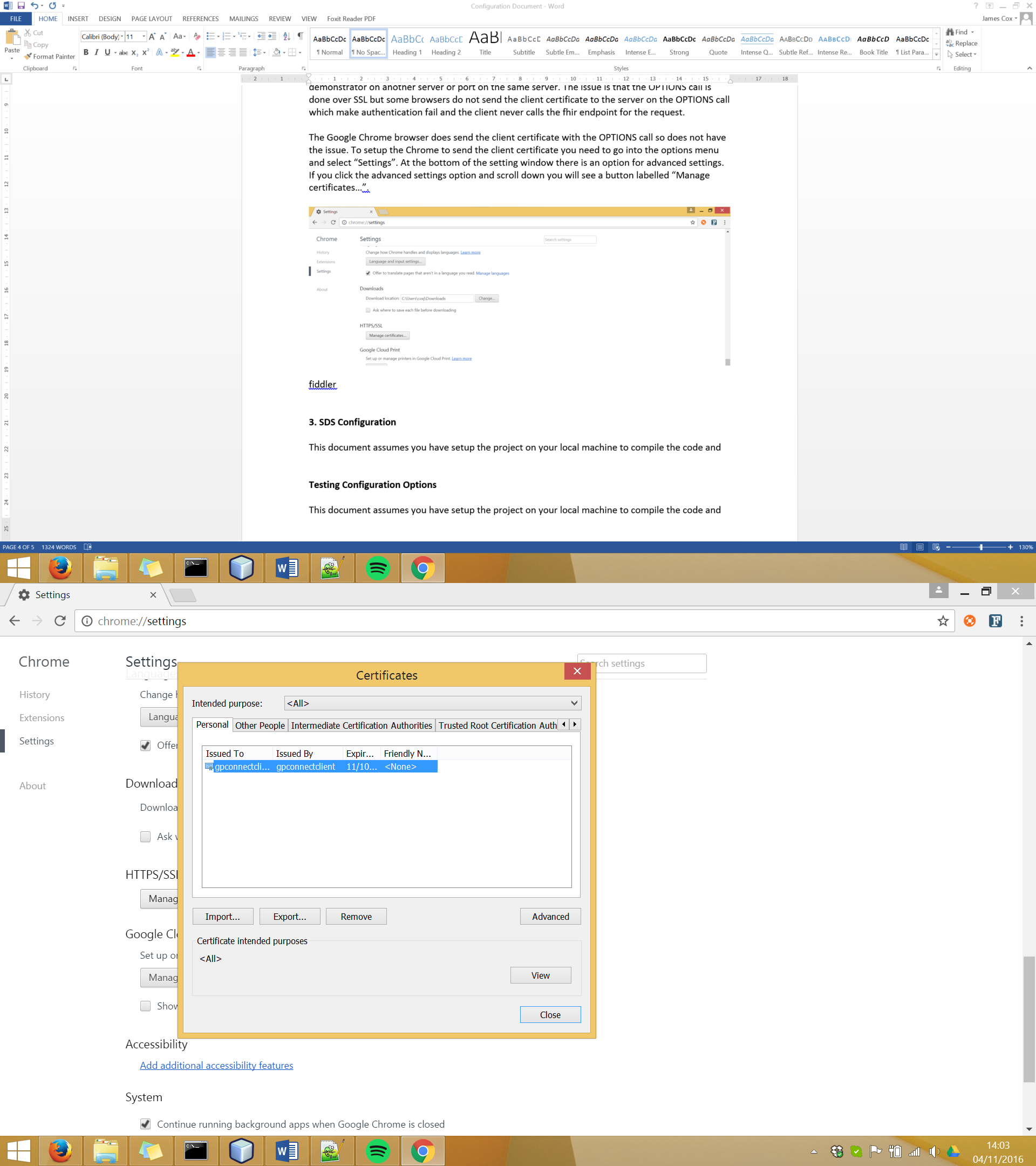


Figure 3 – Google Chrome certificate management window

The other option to get around the CORS issue is to use something such as “fiddler” to intercept the http request and return the CORS headers for the OPTIONS call.

**3. SDS Configuration**

The GP Connect Demonstrator can look up a supplier’s fhir endpoint and ASID using the Spine SDS from the supplier’s ODS code. To do this the demonstrator needs to be configured to talk to the SDS server. If the configuration is wrong or the SDS is not available the demonstrator will use the internal configuration file specified in this document above.

To configure the GP Connect demonstrator to talk to the SDS the following environmental variables need to be configured within the server context. The configuration needs to point towards the SDS server and specify the SSL information required for communication with the SDS.

<Environment name="ldap.context.url" type="java.lang.String" override="false" value="192.168.54.6"/>

<Environment name="ldap.context.port" type="java.lang.Integer" override="false" value="389"/>

<Environment name="ldap.context.useSSL" type="java.lang.Boolean" override="false" value="false"/>

<Environment name="ldap.context.keystore" type="java.lang.String" override="false" value="C:\GPConnectTomcat\ldapKeystore.jks"/>

<Environment name="ldap.context.keystore.pwd" type="java.lang.String" override="false" value="password"/>

<Environment name="ldap.context.keystore.type" type="java.lang.String" override="false" value="JKS"/>

|  |  |
| --- | --- |
| Variable Name | Description |
| ldap.context.url | This contains the URL or IP of the SDS server which the demonstrator should try to connect to. |
| ldap.context.port | This is the port on which the demonstrator will try and connect to the server. |
| ldap.context.useSSL | This allows the demonstrator to communicate with the SDS either over an SSL connection or a non-SSL connection. |
| ldap.context.keystore | This is the absolute path to the JKS (Java Key Store) which contains the SDS server certificate for an SSL connection to the SDS. This will only be used if the “useSSL” variable is set to “true”. |
| ldap.context.keystore.pwd | This is the password for the keystore above. |
| ldap.context.keystore.type | This specifies the keystore type which is being supplied. This is usually “JKS” but other certificate stores can be used. |

**4. Testing Configuration Options**

The latest addition to the GP Connect Demonstrator is a set of testing configuration options. This is to allow a user to change some options for their current GP Connect Demonstrator session. On the main search page there is a new dropdown section labelled “Demonstrator – Testing Configuration”.

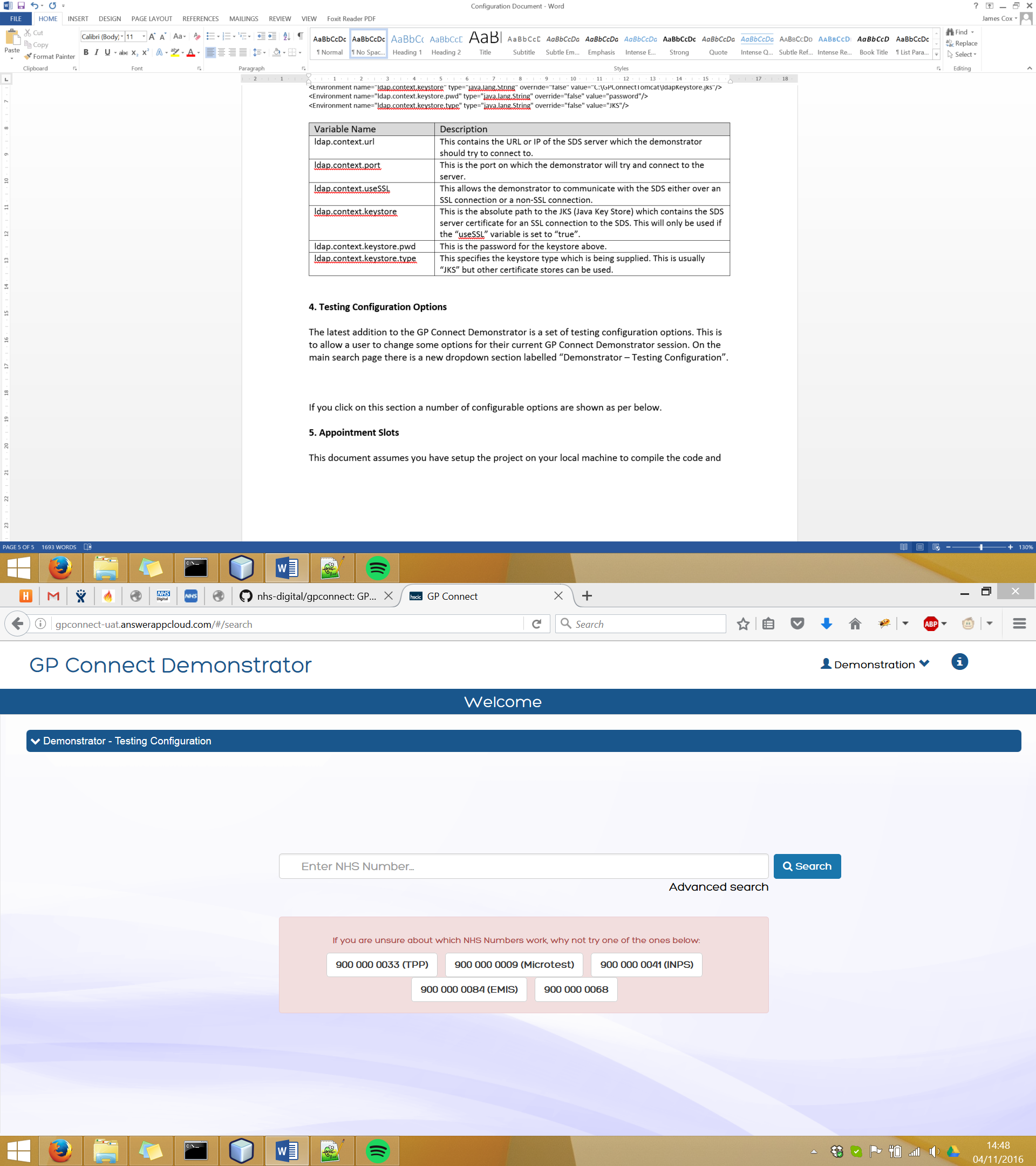


Figure 4 – GP Connect Demonstrator main search page

If you click on this section the panel will open and a number of configurable options for the demonstrator are shown (Figure 5).

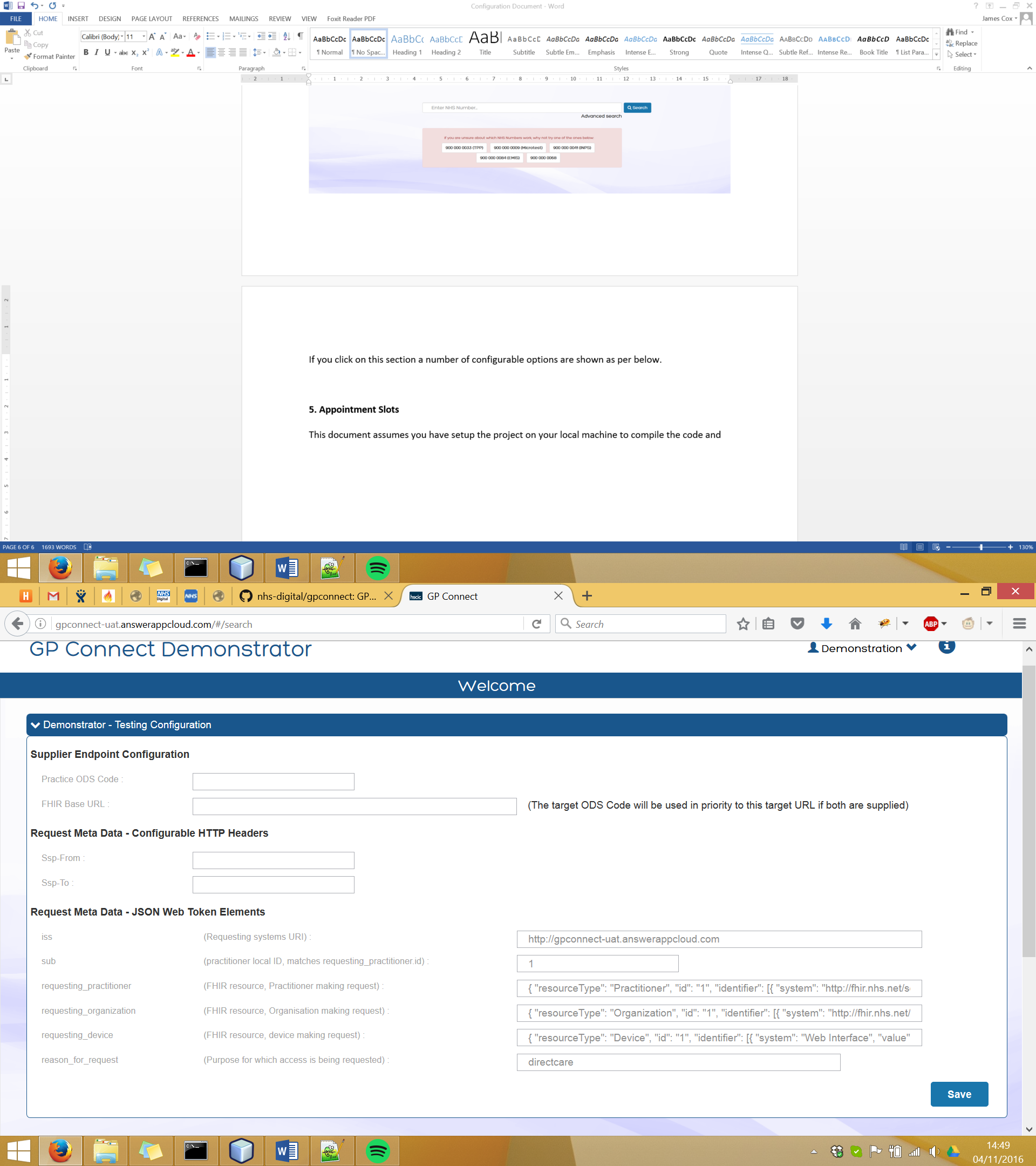


Figure 5 – Demonstrator configuration options window

The configuration is divided up into three main sections. The supplier endpoint configuration, the configurable fhir request headers and the Spine Proxy JWT (JSON Web Token) configurable elements. If no value is entered in any of the sections the default values will be used, if a value is entered that element will be modified in the fhir requests when the user navigates through the demonstrator up until they remove the value or refresh the page which will clear down any entered values.

I will not go into any detail on the fields which can be updated as they are clearly defined within the GP Connect specification. The only element to note is that if a user enters both an endpoint URL and an endpoint ODS code then the system will use the overridden ODS over the URL and will attempt to use the endpoint lookup to find the endpoint URL it is going to use to make the fhir request.