**Emails**

An interface will be provided that will take in the policy object, to get the email address and the user’s full name. This interface will support sending HTML emails. An SMTP account or a service like Sendgrid will need to be setup and integrated with.

For the content of the emails, the same templating engine for policy generation will be used. The interface for use will be described in the document creation section below. A configuration parameter will be used so that the generation will return the compiled HTML, instead of a PDF. HTML templates should be provided that will have content areas to replace, such as name, email address, etc.

**Document Management**

Three main interfaces will be provided for use for the creation, storage, and retrieval of documents.

For storage, S3 will be used, and a wrapper for saving will be provided, taking the file contents and the desired filename/key. A new documents table will be created to keep records of the saved documents and the policy and revision they are tied to.

For retrieving the documents, a wrapper will be created so that the client can request a document, and the backend will retrieve the document from s3 and return it in the request. This is so that the client does not need to interface with S3 directly.

**Document Creation**

HTML templates will need to be provided. The creation wrapper will accept a policy and the desired template to generate the document. Currently, the policy generation function accepts a set group of replacement constants, which are replaced with the data from the user’s policy.

To support having many different policy documents, and to support the generation of emails with this interface, the function will need to be changed to take in a dynamic set of variables that can be set in the template.

The document conversion takes the template and looks for a HTML tag with an attribute called “node”. This node has a string that looks like this “con:variableName”. The variable name portion should be replaced with the wanted variable. So for example, to get the policy owner’s name, the template designer should put in “con:policy.application.fullName”. Then the document conversion tool will loop through all of the nodes set and replace the contents with the correct value.

Below are examples of how to setup an HTML template for use with the variables. A HTML span tag Is normally used for the node. The span tag contents should contain example text so that it is easy to see how it will look while viewing the template before it has been converted. The policy template currently in use will also be attached.

<span node="con:policy.application.fullName">[Full Name]</span>

The node argument will contain con:[variable name]. con: can stay the same, and the portion after the colon will contain a policy object variable. In this case, it is returning the name from the application object. The text inside the span tag should contain a descriptive phrase of what the data will be. This can be anything, as the template generator does not use this, it is only for our internal use.

Another object that is available is coverage. For example, the start date of the coverage can be received by using policy.coverage.start\_date.

<span node="con:policy.coverage.startDate"></span>