\section{Edit .feature files}

Finally, user can start edit the feature file. In feature file, user will need to define background, list edges, scenario outline and scenario. In background, user will need to define the parameters and the distribution, user can also define metavariables, then user will need to define variables are recorded at the end, last user can also define extra conditions for the scenario with \textsl{And} syntax in Cucumber. \\\*\\\*

In the next part, user will need the list the edges of models, this part allow Causcumber to focus on relations of parameters in model define by the user. Next is \textsl{Scenario outline}, in this part user can define changes in parameter and the expected outcome, this require user to provide \textsl{Example}. Finally is \textsl{Scenario}, this is mostly same as \textsl{Scenario outline} but without \textsl{Example}.\\\*\\\*

All the parts mentioned above require user to formatted in a certain syntax, with the interface, the syntax is incorporated into the format of the interface’s design. Starting with background, user need to edit the feature file’s name, then user can start setting the parameters, parameters’ type and parameters’ value. Next user can add meta variables, this part isn’t necessary for feature file, and can be left empty. Next is the recorded variables, this part is similar to the meta variables but will generate different syntax upon create feature file. Last is the for user to add extra condition to the background if needed. This part is structured very similar to the \textsl{.feature} file’s syntax, but with some modification to make those syntax even more plain text.

\begin{figure}[H]

\centering

\includegraphics[width=10cm]{figures/editFeature1Screen.png}\\

\caption{Edit first section of .feature file.}

\label{fig:figure16}

\end{figure}

Next part is to define the edges, this part is mostly identical to the edit \textsl{.dot} function to keep the consistence of the system, the only new addition is a new option allow user to add new edges. This new function is added due to the potential need for user to add extra edges, user can choose to not add new edges if there’s no need for that.\\\*\\\*

After this is scenario outline and scenario, both are also structured like background, where parts of the sentence that require user to edit being left blank. In scenario outline, user will need to define “example”, which is how parameters will change when another parameter is changed in some way. The syntax for example is a table like structure, this is simulated in the interface where user can use the add column and add row to edit the example.

\begin{figure}[H]

\centering

\includegraphics[width=10cm]{figures/editFeature2Screen.png}\\

\caption{Edit scenario outline section of .feature file.}

\label{fig:figure17}

\end{figure}

In scenario, instead of example, it uses “Then” and “And” structure in Cucumber to represent the expected result. The structure is similar to the previous page but with example replaced with the option to add “And”. The user will need to have at least one basic expected result, and if the user wishes to add more expected result, user can use “add And” option to add more.

\begin{figure}[H]

\centering

\includegraphics[width=10cm]{figures/editFeature3Screen.png}\\

\caption{Edit scenario section of .feature file.}

\label{fig:figure18}

\end{figure}

When done, user can just click “Finish” button to return to main menu, then select and run the feature file created.