Appendices J

Graphical User Interface

1. GridBag Layout Manager

```
131             controlsPanel.setLayout(new GridBagLayout());
132
133             GridBagConstraints gc = new GridBagConstraints();
134
135             ////////// FIRST ROW //////////
136             gc.gridx = 0;
137             gc.gridy = 0;
138
139             gc.weightx = 1;
140             gc.weighty = 1;
141
142             gc.fill = GridBagConstraints.NONE;
143             gc.anchor = GridBagConstraints.EAST;
144             gc.insets = new Insets(0, 0, 0, 0);
145             controlsPanel.add(new JLabel("User: "), gc);
146
147             gc.gridx++;
148             gc.anchor = GridBagConstraints.WEST;
149             gc.insets = new Insets(0, 0, 0, 5);
150             controlsPanel.add(userField, gc);
151
152             ////////// NEXT ROW //////////
153             gc.gridx = 0;
154             gc.gridy++;
155
156             gc.weightx = 1;
157             gc.weighty = 1;
158
159             gc.fill = GridBagConstraints.NONE;
160             gc.anchor = GridBagConstraints.EAST;
161             gc.insets = new Insets(0, 0, 0, 0);
162             controlsPanel.add(new JLabel("Password: "), gc);
```

Fig x – Extract from LoginForm class, extract from layoutComponents()

As you can see from fig x, to implement grid bag layout I had to firstly set the layout of the panel in which my form is going to be within, by invoking setLayout() and passing as a parameter a new GridBagLayout object. I then had to declare and initialise a GridBagConstraints object, which is used to define the layout of any components added to the panel.

Once I have a GridBagConstrains object I then invoke methods from the GridBagConstraints

class to set the constraints of any components added. The basic constraints are gridx() and gridy() which defines what position I am going to add a component on the screen, and gridwidth() and gridheight() determines what size the cell will take up on the screen.

2. Model-View-Controller

```
34      private ListPanel listsPanel;
```

Fig x – Extract from HomeForm class – declaration of ListPanel object

```
86          try {
87              this.updateAgreementsList(client.getUserAgreements());
88          } catch (RemoteException ex) {
89              Logger.getLogger(HomeForm.class.getName()).log(Level.SEVERE, null, ex);
90          }
91
92          listsPanel.setTableListener(new TableListener() {
93              @Override
94              public void rowSelected(Object agreement) {
95                  if(agreement instanceof TenancyInterface) {
96                      TenancyInterface tenancy = (TenancyInterface) agreement;
97  //                      TenancyDetailsForm tenancyForm = new TenancyDetailsForm();
98  //                      tenancyForm.setClient(client);
99  //                      tenancyForm.setTenancy(tenancy);
100                 } else if(agreement instanceof LeaseInterface) {
101                     LeaseInterface lease = (LeaseInterface) agreement;
102 //                      LeaseDetailsForm leaseForm = new LeaseDetailsForm();
103 //                      leaseForm.setClient(client);
104 //                      leaseForm.setLease(lease);
105                 } else if(agreement instanceof ContractInterface) {
106                     ContractInterface contract = (ContractInterface) agreement;
107 //                      ContractDetailsForm contractForm = new ContractDetailsForm();
108 //                      contractForm.setClient(client);
109 //                      contractForm.setContract(contract);
110                 }
111             }
112         });
```

Fig x – Extract from HomeForm class – HomeForm constructor

```
29      private TableListener tableListener;
```

Fig x – Extract from ListsPanel class, initialising TableListener field (action listener)

```
78      public void setTableListener(TableListener tenListener) {
79          this.tableListener = tenListener;
80      }
```

Fig x – Extract from ListsPanel class, setTableListener()

As you can see from fig x, fig x, fig x and fig x, I am ensuring the GUI makes use of MVC, by assigning the JPanels within any main frame with listeners, so if anything occurs within a panel

(such as the listsPanel for the HomeForm shown above), instead of the listsPanel invoking a method from the HomeForm to notify the HomeForm of any change that has occurred within the panel, the panel is passed an action listener which listens to see if any action has been performed, and if so carries out a function within the main frame.

This ensures that the view (ListsPanel) does not know about the controller (HomeForm) and only interacts with the action listener that was passed to the ListsPanel by the HomeForm, meaning that the HomeForm can be independent, to the