Working with the Demo Applications

The Instant Demo

- template.jsp/template.master
- Screen Name: Login
- Screen Name: InsuranceMenu
- Screen Name: BrowseCustomers

Note:
When using a replay file, it is possible to navigate from one screen to the next by pressing ENTER in each web screen. Clicking on buttons which execute paths will work only if the replay file contains the relevant path navigation recorded and the session screen is in the relevant starting point to execute the path. If this is not the case, the path will fail and the user will receive an error from the web application.

When using transformations it is important to notice the order in which the transformations appear in the screen editor (**Transformations** tab). Transformations can change the way ApplinX perceives a host screen. If, for example, the first transformation hides a certain text on the screen and the second transformation searches for that text pattern in order to replace it with a link then the second transformation will never find a match for that pattern since it has been removed by the first transformation. So always keep in mind, the order of the transformations in the screen editor will be the order in which they will be executed.

template.jsp/template.master

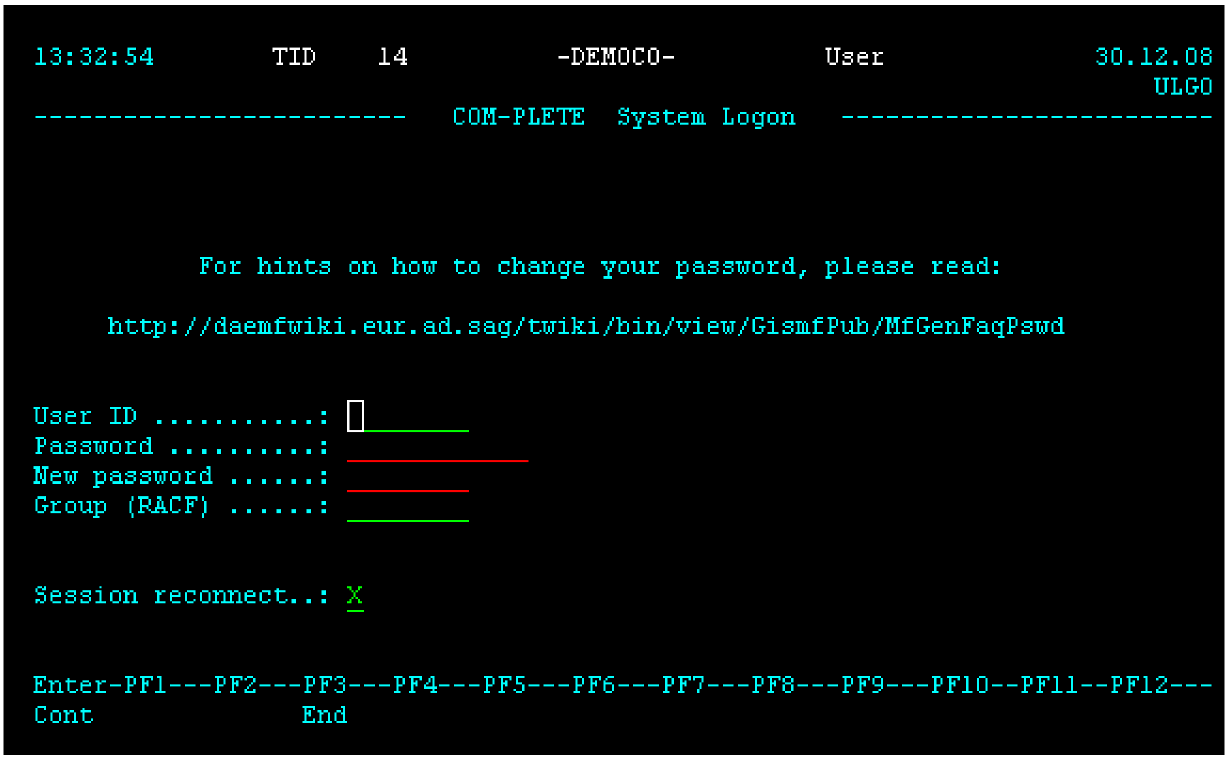
Refer to *Customizing the Default Template* in the *Web Application Development* documentation for a detailed explanation on how to change the default template.

- The host keys relevant for this screen are displayed on the left of the screen. This is defined in the *template.jsp/template.master* (.NET web apps) file in the host key control. The host key control displays all the host keys for the current host screen. It is possible to define whether they should appear horizontally or vertically , in the form of buttons, links or a pre-defined template (see "gx:GXHostKeysControl" in .NET web apps or "gx:hostKeys" tag in JSP applications). The default host key style is defined in the *css/styles_instant.css* style sheet in the "gx_hky" class.

- The Page footer section contains links that perform various actions such as logging off, refreshing the page, submitting a terminal key and so on. These links can be defined in the *template.jsp/template.master* file (look for "pgfooter" control in the instant demo app).

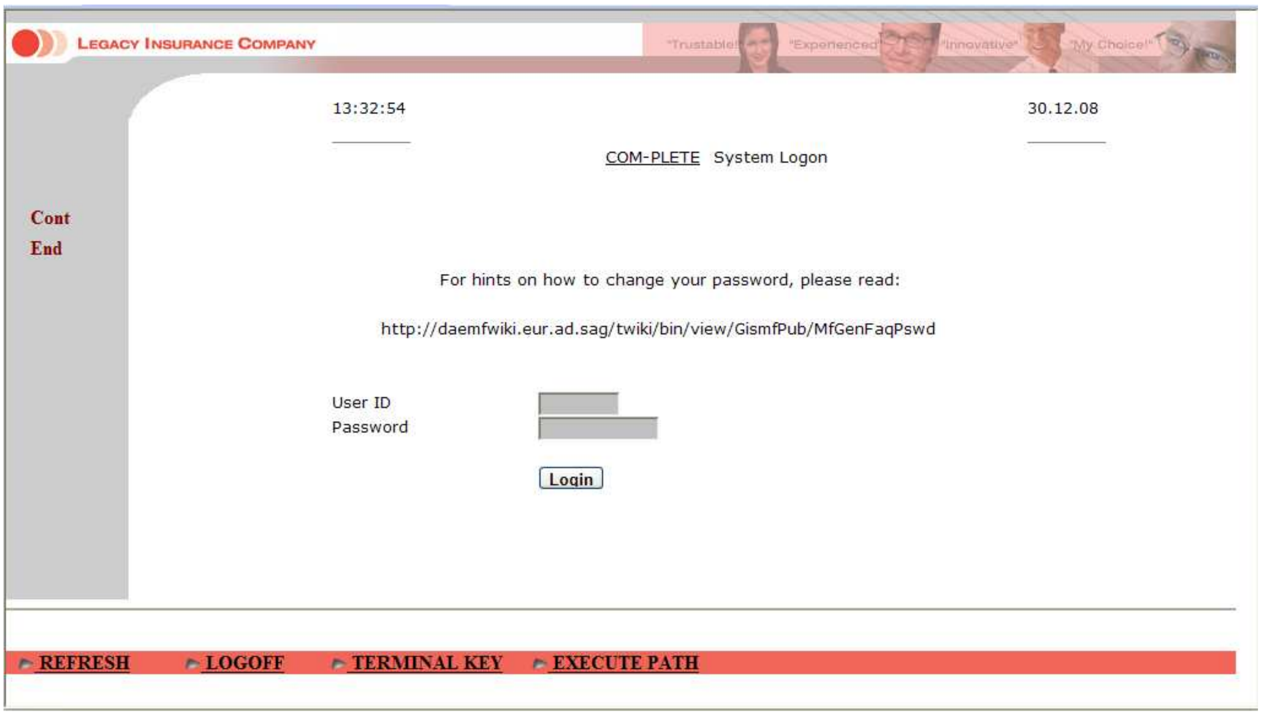
Screen Name: Login

Host Screen



```
13:32:54         TID     14           -DEMOCO-           User             30.12.08
                                                                              ULGO
-------------------------   COM-PLETE  System Logon   -------------------------



           For hints on how to change your password, please read:

     http://daemfwiki.eur.ad.sag/twiki/bin/view/GismfPub/MfGenFaqPswd


User ID ...........: ________
Password ..........: ____________
New password ......: ________
Group (RACF) ......: ________


Session reconnect..: X


Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
Cont                 End
```

Screen in Web Application



LEGACY INSURANCE COMPANY

13:32:54 30.12.08

COM-PLETE System Logon

For hints on how to change your password, please read:

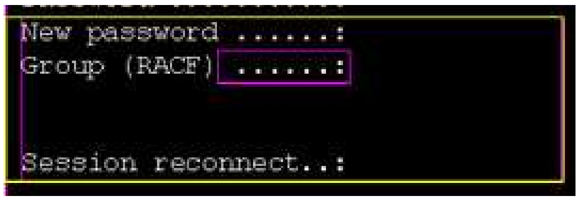
http://daemfwiki.eur.ad.sag/twiki/bin/view/GismfPub/MfGenFaqPswd

User ID

Password

Note:
The sections that are referred to below can be found in the *Designing and Developing an Application* documentation.

- Identify the screen using the "-DEMOCO-" and "COM-PLETE System Logon" texts. Refer to *Define Identifiers*.
- Map the **Message**, **UserID**, **Password** and **Reconnect** fields. Refer to *Map Fields*.
- The "HideTextTODel" transformation hides text in a defined area. Refer to *Transforming a Text Pattern to Text* and select the "Hide" option.



```
New password ......:
Group (RACF) ......:


Session reconnect..:
```

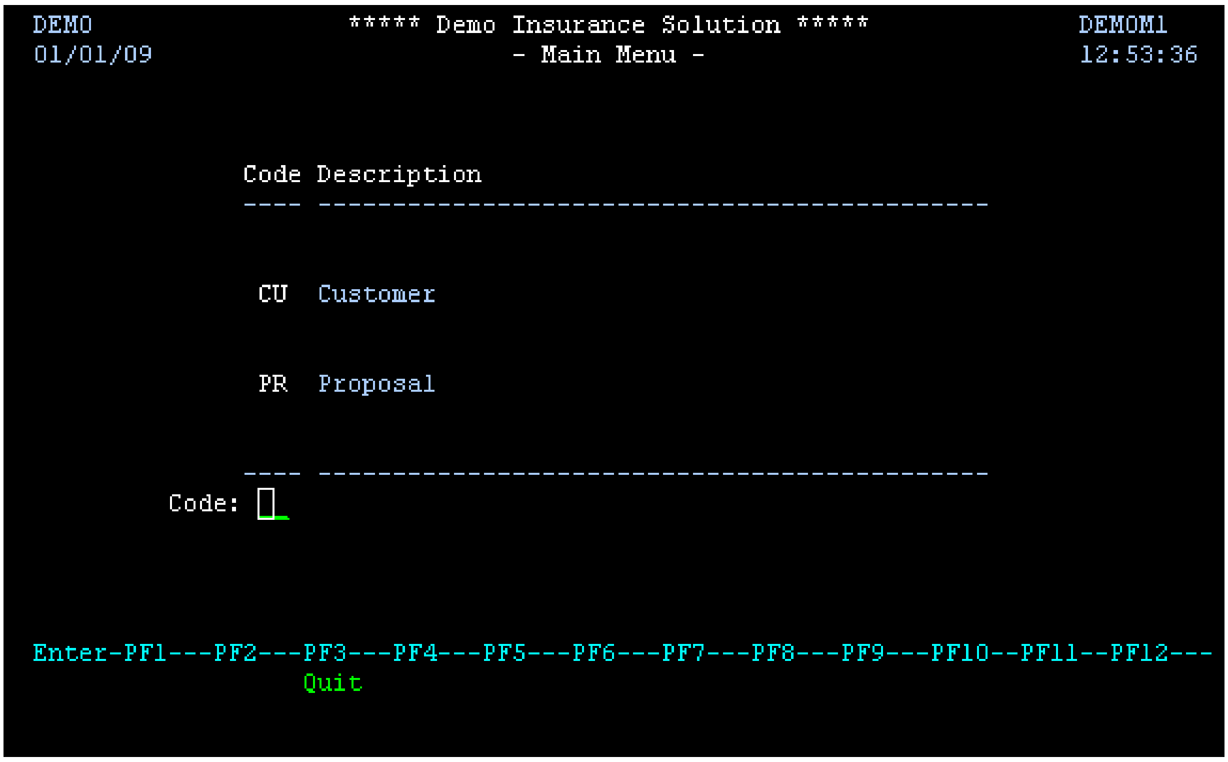
- The "RemoveDots" transformation removes dots followed by a colon ("...:"). Refer to *Transforming a Text Pattern to Text*.
- The "HideInputField" transformation hides input fields in a defined area. Refer to *Transforming an Input Field to a Text Field* and select the "Hide" option.
- The "HideText" transformation hides text in different areas in the screen. Refer to *Transforming a Text Pattern to Text* and select the "Hide" option.
- The "DashesToLine" transformation changes all the dashes in the screen to lines. Refer to *Transforming a Repeating Characters Pattern to a Line*.

- The "AddLoginButton" transformation adds a button used to login. Clicking on this button executes the "login" path procedure. Refer to *Transforming a Text Pattern to a Button*.

  The "login" path procedure receives the user ID and password as inputs.

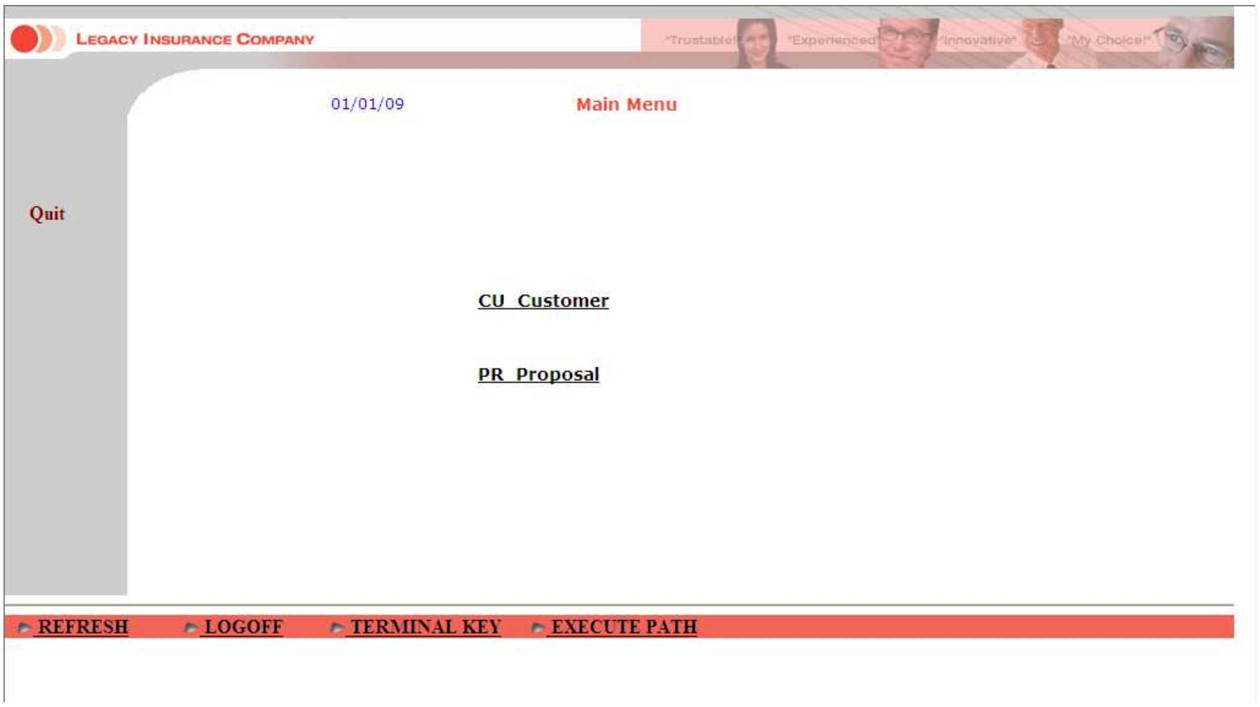
Screen Name: InsuranceMenu

Host Screen



```
DEMO                    ***** Demo Insurance Solution *****                DEMOM1
01/01/09                            - Main Menu -                          12:53:36


                Code Description
                ---- ---------------------------------------------


                 CU  Customer


                 PR  Proposal


                ---- ---------------------------------------------
          Code:  _



Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
                  Quit
```

Screen in Web Application



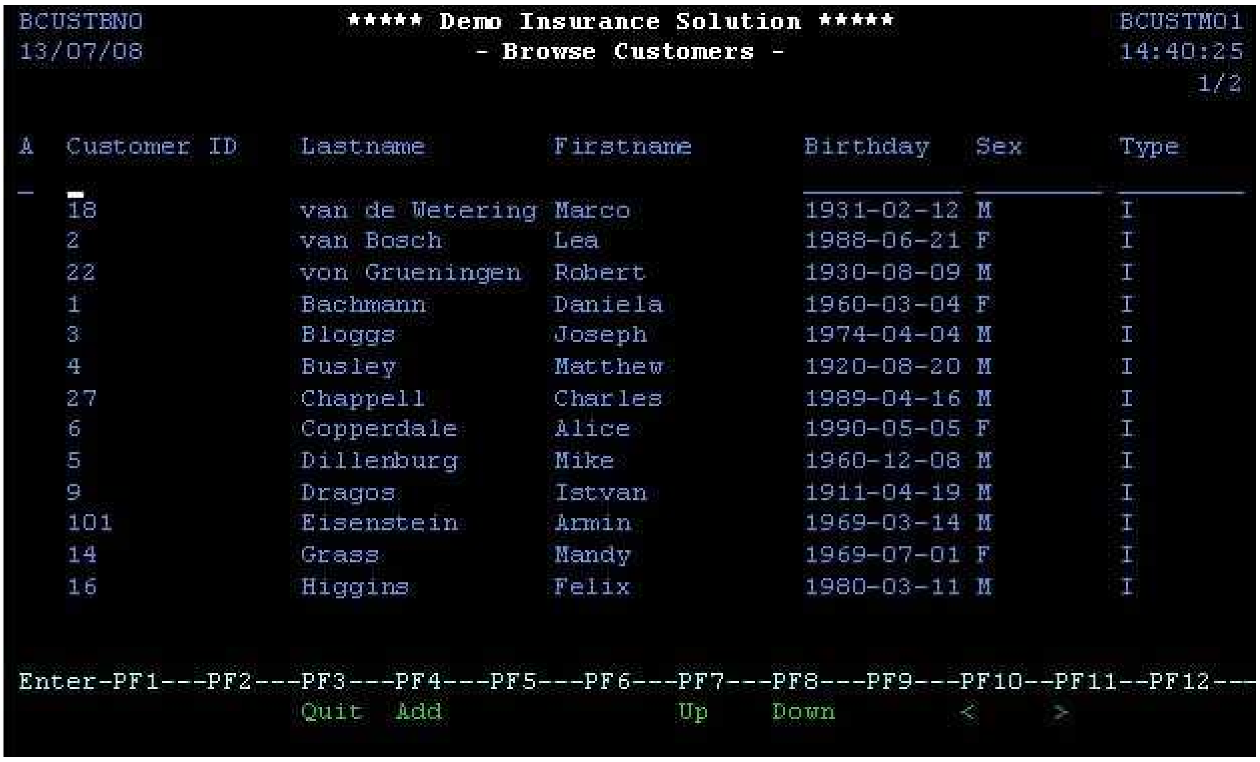
Note:
The sections that are referred to below can be found in the *Designing and Developing an Application* documentation.

- Identify the screen using the "DEMO" and "DEMOM" texts. Refer to *Define Identifiers*.
- Map the **Code**, **Time** and **Message** fields. Refer to *Map Fields*.
- The "CleanMenuScreen" transformation hides text in a number of defined areas in the screen. Refer to *Transforming a Text Pattern to Text* and select the "Hide" option.
- The "AllGroup" screen group includes all the screens that include in row 1, column 23 the content "*****Demo Insurance Solution*****". The "AllGroup" screen group includes various transformations:
  - Removing the first line in the screen (the "RemoveFirstLine" transformation). Refer to *Transforming a Text Pattern to Text* and select the "Hide" option.
  - Formatting the screen name (the "FormatScreenNameHeader" transformation).
  - Removing dots that are followed by a colon "...:" (the "RemoveDots" transformation). Refer to *Transforming a Text Pattern to Text*.
  - Hiding the time that appears in the top right corner (the "HideText" transformation). Refer to *Transforming a Text Pattern to Text* and select the "Hide" option.
- The "MenusGroup" screen group includes all screens that include text: "code", and text "menu". This screen group includes a transformation which transforms menus to links (the "MenuToLinks" transformation). Refer to *Transforming a Menu to Hyperlinks*.

- Associate the "AllGroup" and "MenusGroup" screen groups to this screen.
- Refer to the template section above to see how to implement placing the host keys on the left of the screen, and placing buttons in the footer.

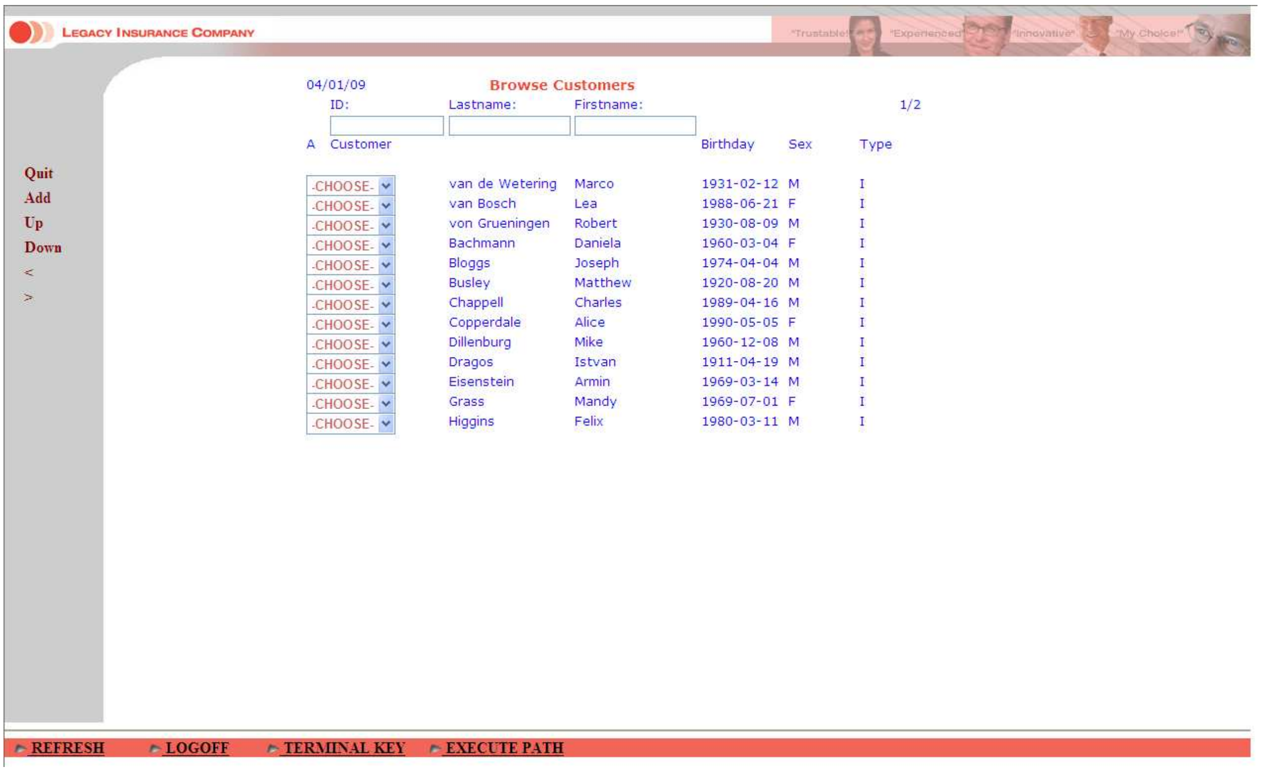
Screen Name: BrowseCustomers

Host Screen



```
BCUSTBNO                ***** Demo Insurance Solution *****              BCUSTM01
13/07/08                        - Browse Customers -                     14:40:25
                                                                              1/2

A  Customer ID    Lastname         Firstname        Birthday   Sex       Type

_  _                                                __________ ________  ________
   18             van de Wetering  Marco            1931-02-12 M         I
   2              van Bosch        Lea              1988-06-21 F         I
   22             von Grueningen   Robert           1930-08-09 M         I
   1              Bachmann         Daniela          1960-03-04 F         I
   3              Bloggs           Joseph           1974-04-04 M         I
   4              Busley           Matthew          1920-08-20 M         I
   27             Chappell         Charles          1989-04-16 M         I
   6              Copperdale       Alice            1990-05-05 F         I
   5              Dillenburg       Mike             1960-12-08 M         I
   9              Dragos           Istvan           1911-04-19 M         I
   101            Eisenstein       Armin            1969-03-14 M         I
   14             Grass            Mandy            1969-07-01 F         I
   16             Higgins          Felix            1980-03-11 M         I


Enter-PF1---PF2---PF3---PF4---PF5---PF6---PF7---PF8---PF9---PF10--PF11--PF12---
                  Quit  Add                 Up    Down        <     >
```

Screen in Web Application



04/01/09 Browse Customers 1/2

ID: Lastname: Firstname:

A	Customer		Birthday	Sex	Type
-CHOOSE-	van de Wetering	Marco	1931-02-12	M	I
-CHOOSE-	van Bosch	Lea	1988-06-21	F	I
-CHOOSE-	von Grueningen	Robert	1930-08-09	M	I
-CHOOSE-	Bachmann	Daniela	1960-03-04	F	I
-CHOOSE-	Bloggs	Joseph	1974-04-04	M	I
-CHOOSE-	Busley	Matthew	1920-08-20	M	I
-CHOOSE-	Chappell	Charles	1989-04-16	M	I
-CHOOSE-	Copperdale	Alice	1990-05-05	F	I
-CHOOSE-	Dillenburg	Mike	1960-12-08	M	I
-CHOOSE-	Dragos	Istvan	1911-04-19	M	I
-CHOOSE-	Eisenstein	Armin	1969-03-14	M	I
-CHOOSE-	Grass	Mandy	1969-07-01	F	I
-CHOOSE-	Higgins	Felix	1980-03-11	M	I

- Identify the screen using the "BCUSTBN0" and "BCUSTBN1" texts. Refer to *Define Identifiers* in the *Designing and Developing an Application* documentation.
- Refer to the Fields tab to see the mapped fields.
- Refer to the template section above to see how to implement placing the host keys on the left of the screen, and placing buttons in the footer.

The Composite Demo

- template.jsp/template.master
- Screen Name: Login
- Screen Name: Browse Proposals
- Screen Name: ModifyProposal1
- Screen Name: BrowseCustomers1

Note:
When using a replay file, it is possible to navigate from one screen to the next by pressing ENTER in each web screen. Clicking on buttons which execute paths will work only if the replay file contains the relevant path navigation recorded and the session screen is in the relevant starting point to execute the path. If this is not the case, the path will fail and the user will receive an error from the web application.

template.jsp/template.master

Refer to *Customizing the Default Template* in the *Web Application Development* documentation for a detailed explanation on how to change the default template.

JSP

In this template we use these panels (that can be overridden by specific pages):

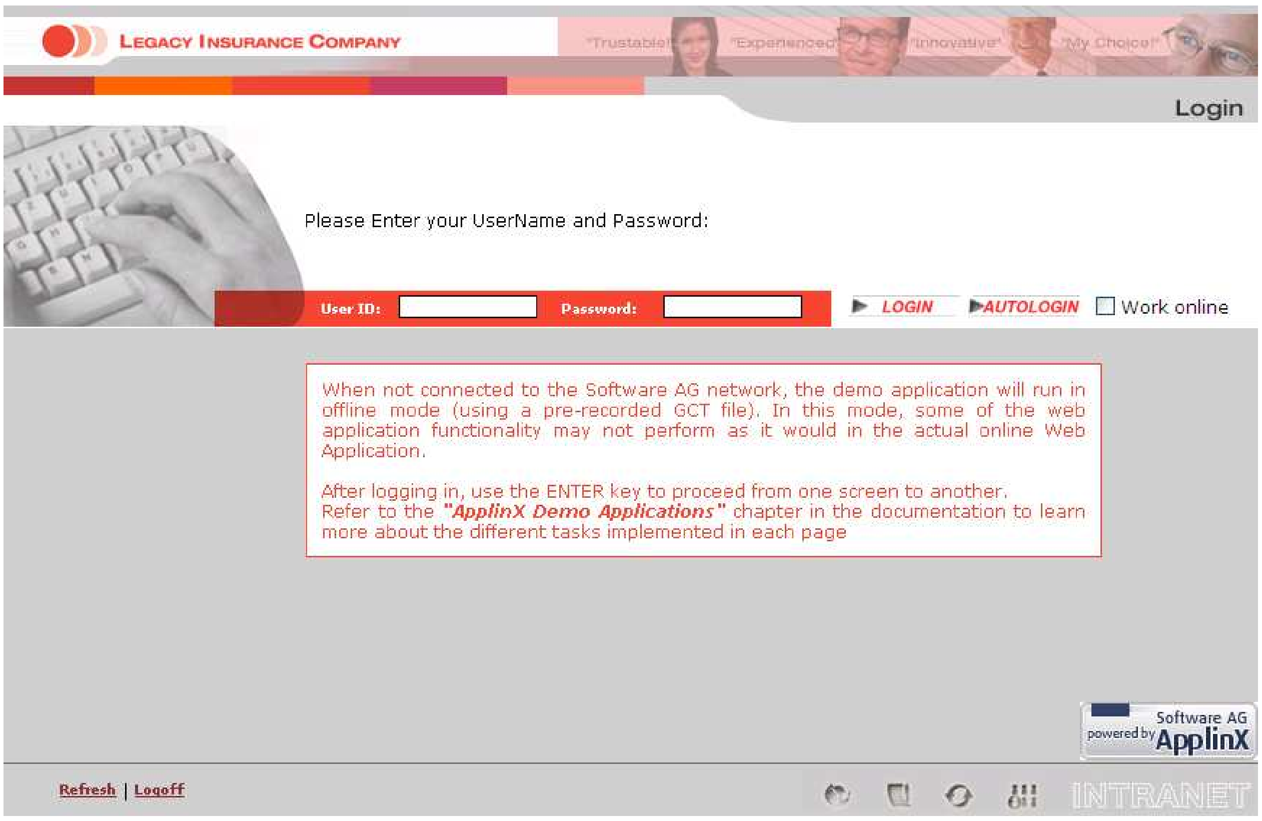
- CssPanel: holds the css links (has a default in the template).
- JSPanel: holds the JavaScript code and JavaScript file references.
- HeaderImagePanel: by design, an image in the top-right corner changes according to the data.
- MenuPanel: holds the menu that is included in the pages. In some specific pages it is empty, therefore the menu is not displayed.
- GXPagePlaceHolder: holds the data for the page.

.NET

In this template we use these panels (that can be overridden by specific pages):

- CssPanel: holds the css links (has a default in the template).
- JSPanel: holds the JavaScript code and JavaScript file references.
- HeaderImagePanel: by design, an image in the top-right corner changes according to the data.
- MenuPanel: holds the menu that is included in the pages. In some specific pages it is empty, therefore the menu is not displayed.
- GXPagePlaceHolder: holds the data for the page.

Screen Name: Login



Please Enter your UserName and Password:

When not connected to the Software AG network, the demo application will run in offline mode (using a pre-recorded GCT file). In this mode, some of the web application functionality may not perform as it would in the actual online Web Application.

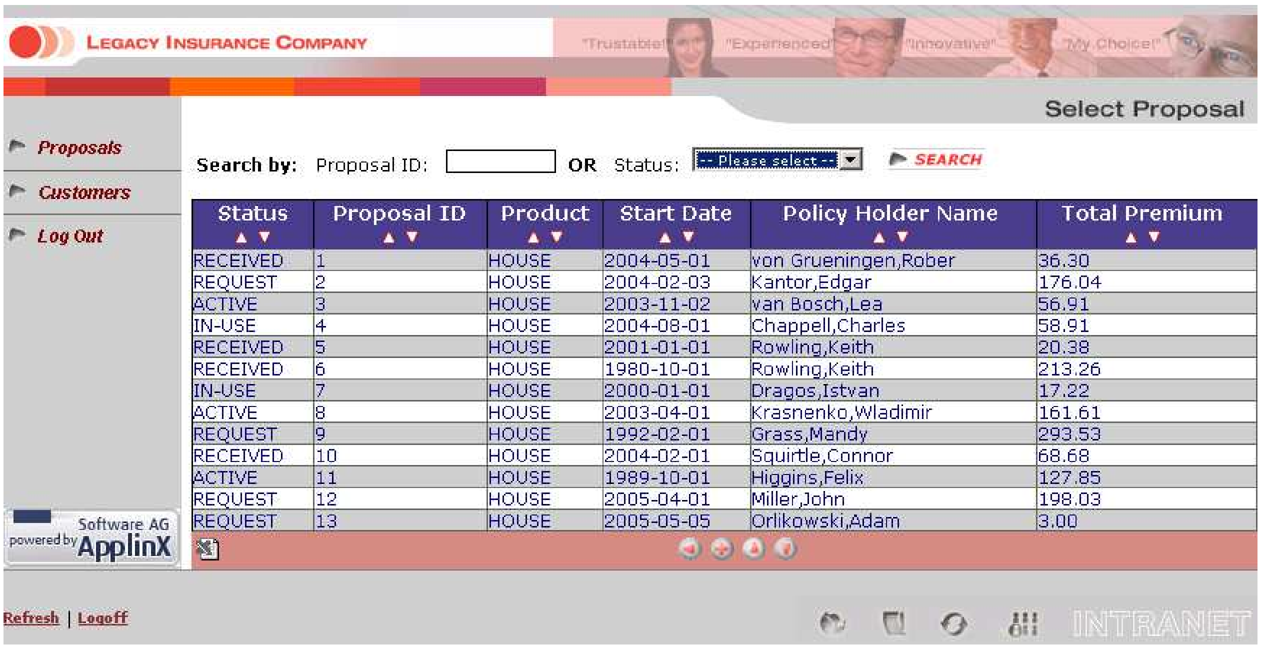
After logging in, use the ENTER key to proceed from one screen to another.
Refer to the **"ApplinX Demo Applications"** chapter in the documentation to learn more about the different tasks implemented in each page

Tasks

Refer to the following sections in the *Web Application Development* documentation:

- *Building an External Login Page*
- *Controlling the Connection Properties from Code*

Screen Name: Browse Proposals



Search by: Proposal ID: OR Status: -- Please select --

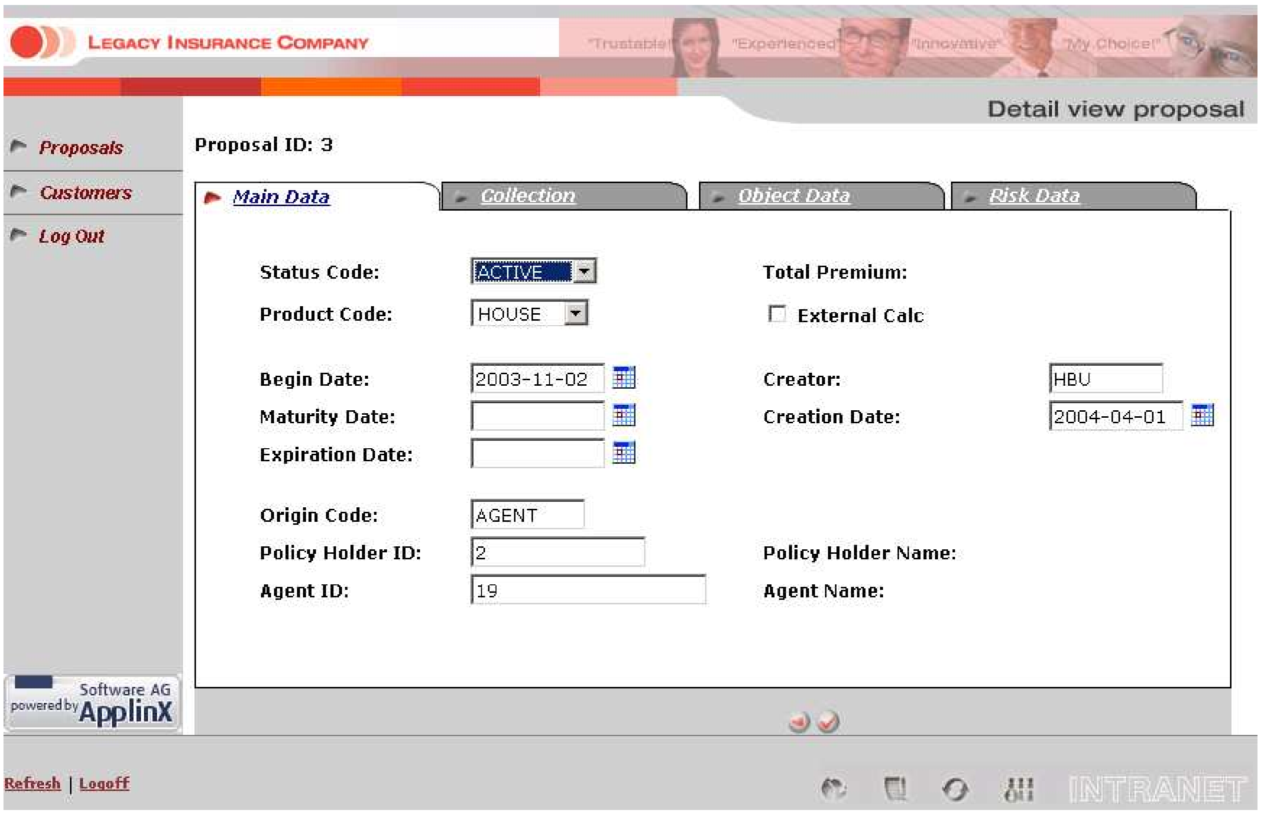
Status	Proposal ID	Product	Start Date	Policy Holder Name	Total Premium
RECEIVED	1	HOUSE	2004-05-01	von Grueningen,Rober	36.30
REQUEST	2	HOUSE	2004-02-03	Kantor,Edgar	176.04
ACTIVE	3	HOUSE	2003-11-02	van Bosch,Lea	56.91
IN-USE	4	HOUSE	2004-08-01	Chappell,Charles	58.91
RECEIVED	5	HOUSE	2001-01-01	Rowling,Keith	20.38
RECEIVED	6	HOUSE	1980-10-01	Rowling,Keith	213.26
IN-USE	7	HOUSE	2000-01-01	Dragos,Istvan	17.22
ACTIVE	8	HOUSE	2003-04-01	Krasnenko,Wladimir	161.61
REQUEST	9	HOUSE	1992-02-01	Grass,Mandy	293.53
RECEIVED	10	HOUSE	2004-02-01	Squirtle,Connor	68.68
ACTIVE	11	HOUSE	1989-10-01	Higgins,Felix	127.85
REQUEST	12	HOUSE	2005-04-01	Miller,John	198.03
REQUEST	13	HOUSE	2005-05-05	Orlikowski,Adam	3.00

Tasks

Refer to the following sections in the *Web Application Development* documentation:

- *Activating an Application Map from a Menu*
- *Creating Designed Web Pages*
- *Creating a Button / Hyperlink for Submitting a Host Key*
- *Using Web Application Controls in Generated Pages*
- *Exporting Data to an MS Office Application (Excel, Word)*
- *Adding the Sorting Capability to a Screen-Based Table*
- *Creating a Page with a Table*

Screen Name: ModifyProposal1



Proposal ID: 3

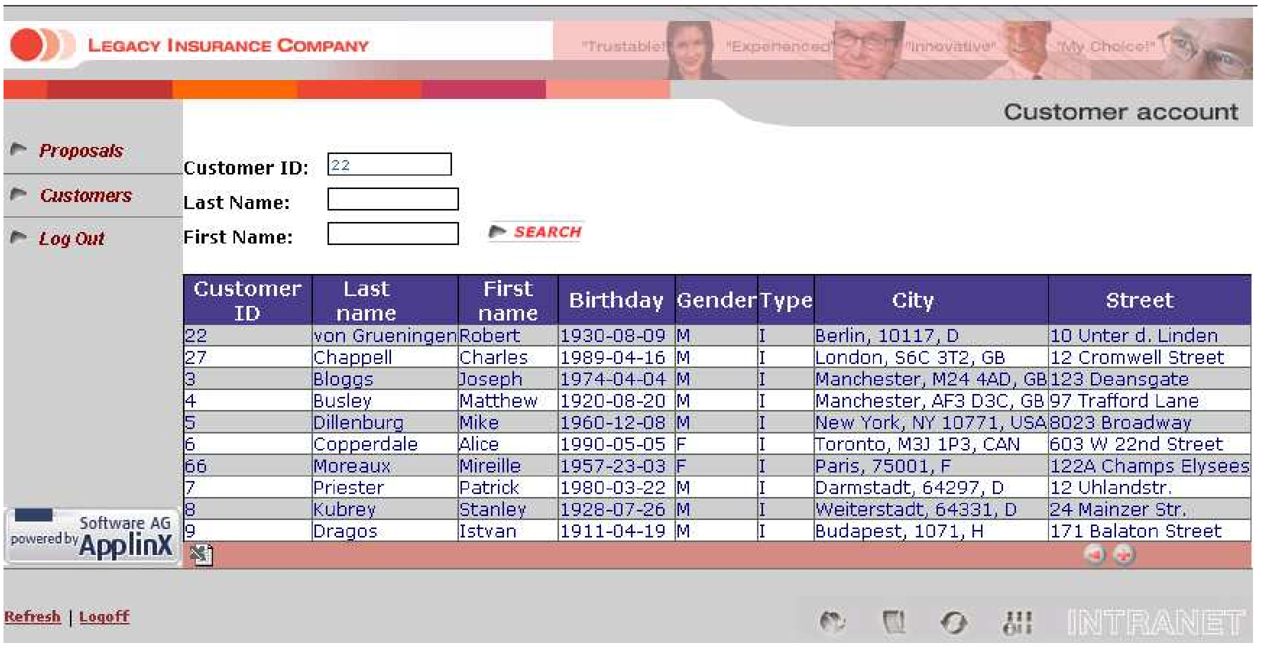
Status Code:	ACTIVE	Total Premium:	
Product Code:	HOUSE	External Calc	
Begin Date:	2003-11-02	Creator:	HBU
Maturity Date:		Creation Date:	2004-04-01
Expiration Date:			
Origin Code:	AGENT		
Policy Holder ID:	2	Policy Holder Name:	
Agent ID:	19	Agent Name:	

Tasks

Refer to the following sections in the *Web Application Development* documentation:

- *Collecting Data from Multiple Host Screens*
- *Creating a Button / Hyperlink for Executing a Path Procedure*
- *Updating Data in Multiple Host Screens*
- *Activating an Application Map from a Menu*
- *Creating Designed Web Pages*
- *Creating a Button / Hyperlink for Submitting a Host Key*
- *Using Web Application Controls in Generated Pages*
- *Map Fields* (identify dynamic fields by their leading label)

Screen Name: BrowseCustomers1



Customer ID:	22
Last Name:	
First Name:	

Customer ID	Last name	First name	Birthday	Gender	Type	City	Street
22	von Grueningen	Robert	1930-08-09	M	I	Berlin, 10117, D	10 Unter d. Linden
27	Chappell	Charles	1989-04-16	M	I	London, S6C 3T2, GB	12 Cromwell Street
3	Bloggs	Joseph	1974-04-04	M	I	Manchester, M24 4AD, GB	123 Deansgate
4	Busley	Matthew	1920-08-20	M	I	Manchester, AF3 D3C, GB	97 Trafford Lane
5	Dillenburg	Mike	1960-12-08	M	I	New York, NY 10771, USA	8023 Broadway
6	Copperdale	Alice	1990-05-05	F	I	Toronto, M3J 1P3, CAN	603 W 22nd Street
66	Moreaux	Mireille	1957-23-03	F	I	Paris, 75001, F	122A Champs Elysees
7	Priester	Patrick	1980-03-22	M	I	Darmstadt, 64297, D	12 Uhlandstr.
8	Kubrey	Stanley	1928-07-26	M	I	Weiterstadt, 64331, D	24 Mainzer Str.
9	Dragos	Istvan	1911-04-19	M	I	Budapest, 1071, H	171 Balaton Street

Tasks

Refer to the following sections in the *Web Application Development* documentation:

- *Activating an Application Map from a Menu*
- *Creating Designed Web Pages*
- *Collecting Data from Multiple Host Screens*
- *Creating a Button / Hyperlink for Submitting a Host Key*
- *Using Web Application Controls in Generated Pages*
- *Adding the Sorting Capability to a Screen-Based Table*
- *Exporting Data to an MS Office Application (Excel, Word)*
- *Customizing the Table's Display*

The SOA Demo

.NET and JSP implementation can be done in many ways. This demo includes examples of one way to implement the tasks. For example, in one of the tasks below, we demonstrate how to display data collected by the procedure client. To do this we use the framework. In .NET, this can also be implemented using .NET datagrid.

The following lists the different typical tasks that we have demonstrated in the SOA application included with the ApplinX installation. Next to each task, the entities involved with this task are detailed. This enables you to further explore the configuration of each of these entities.

1. Identify and define host screens in the ApplinX repository. Refer to *Screens* in the *Designing and Developing an Application* documentation.
   - INS_Browse_Customers
   - INS_Customer_Details_1
   - INS_Customer_Details_2
   - INS_Customer_Details_3
   - DEMOCO_Login
   - DEMOCO_Menu
   - DEMOCO_Splash
   - E_Environment
   - INS_Menu
2. Create a connection pool with a connection information set (for offline use). Refer to *Connection Pools* and *Connection Information Sets* in the *Designing and Developing an Application* documentation.
   1. Set the pool size as required.
   2. Set a connection information set when required (SOADemo includes connection information sets that allow the application to run in offline mode).
   3. Set navigation rules:
      1. Create an initialization path for the pool that initializes each new connection in the pool. This path should finish in the "Initial screen", otherwise the initialization will fail.
      2. Set the initial screen a connection should wait on.
      3. Create and set a recycle path to recycle connections in the pool.
      4. Create and set a termination path for properly terminating a connection in the pool.

   As you may have noticed, the SOADemo repository contains three connection pools. This is for the sole reason of enabling the application to retrieve results in offline environments (that do not have connectivity to Software AG’s network). All three pools share the same setting (pool size and navigation settings), but differ in their connection information sets. To enable offline execution of the application, each procedure has been executed with its own replay file (set in its connection information set), whereas, online execution requires no replay file at all.
3. Collect data from multiple screens. Refer to *Collecting Data from Multiple Host Screens* in the *Web Application Development* documentation.
   1. Create a path procedure which collects table data (refer to the Browse_All_Customers path procedure). Refer to *Creating a Path Procedure* in the *Designing and Developing an Application* documentation.

        1. Assign a connection pool to start from the relevant screen.
        2. Create a While loop that checks if an "End of data" message appears on the screen, marking the table’s end.
        3. For every table instance: Collect the tabular host screen data and page down for the next instance.

    2. Create a path procedure which collect data from multiple screens (refer to the GetCustomerByID path procedure). Refer to *Creating a Path Procedure* in the *Designing and Developing an Application* documentation.
        1. Assign a connection pool to start from the relevant screen.
        2. Add an input to allow the user to specify which customer details they wish to retrieve.
        3. Navigate to the Details screens and map host fields to the procedures’ output.
    3. Assign procedures to procedure groups. Refer to *Assigning a Procedure to a Procedure Group* in the *Designing and Developing an Application* documentation.
    4. Generate a procedure client from the procedure group (refer to DemoINS procedure group in the demo repository). Refer to *Procedure Clients* in the *Designing and Developing an Application* documentation.

4. Bind web controls to procedure response (output). Refer to *Binding Procedure Outputs to an ApplinX Framework Based Web Page* in the *Designing and Developing an Application* documentation.
    1. For each procedure, create a JSP/ASPX page with controls which match the procedure output.
    2. Fill the form (using gx_fillForm).