**Insurance company**

**Info 445 - Event Programming and Graphical Interface**

**Sommaire**

* Introduction ……….……………………….….2
* fonctionnalité……………..…..……………….3
* Mobile modèle………………….……..………4
* Design pattern……….…………………………6
* Conclusion …………………………..…………..9

**Introduction**

Dans ce problème nous nous intéressons au développement d’une application d’une compagnie d’assurance.

Cette application doit faciliter le travail de l’employer, la rapidité de transmission entre une assurance et l’autre.

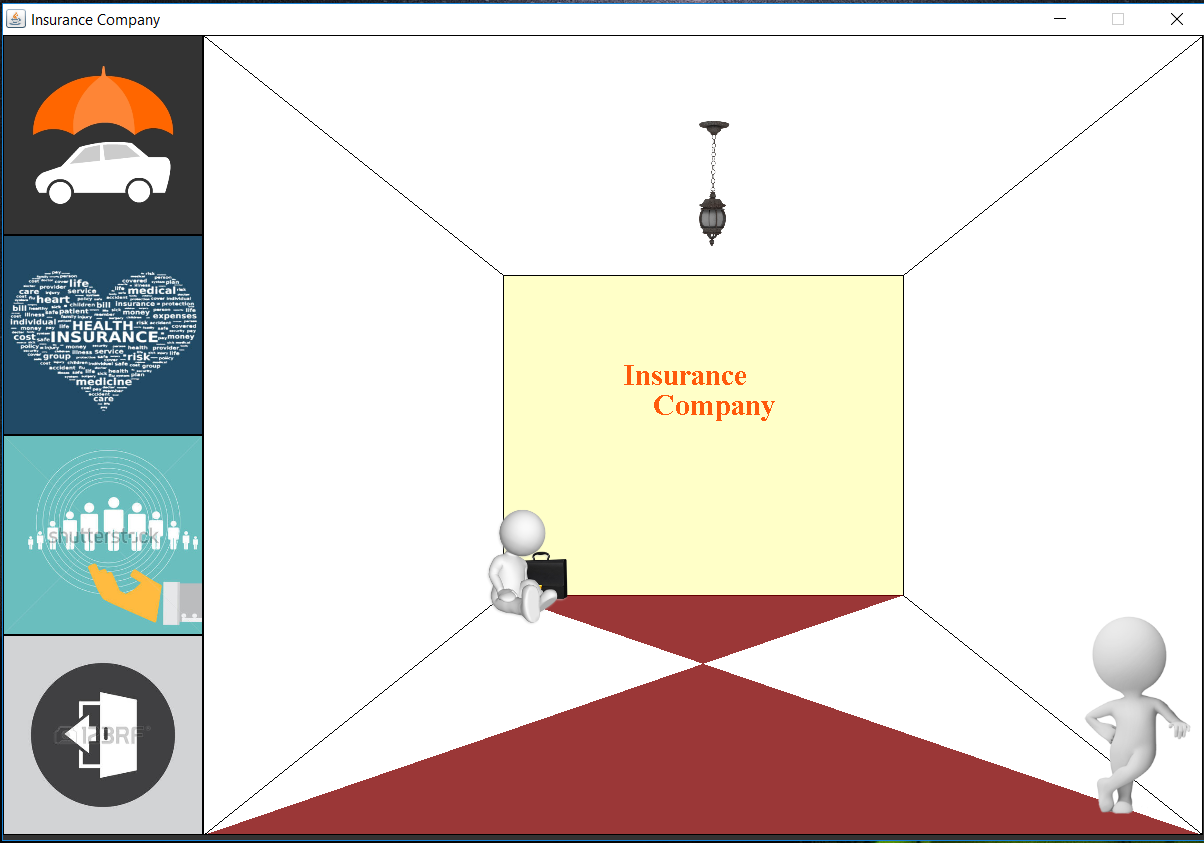
Un autre objectif de cette application est de changer le rythme rigide qui manque de souplesse en utilisant un mobile modèle qui utilise plusieurs processus.

D’autre part l’utilisation des ‘design pattern’ pour la simplicitée du code et améliorer la performance du l’application.

**Fonctionnalité** :

Cette application permet à l’employé d’ajouter des assurances sur la vie, voiture ou santé. Et elle permet au client de voire ces anciennes assurances.

L’employé doit faire une authentification ensuite choisi entre les assurances par un ‘drag and drop’ de l’image convenable.



‘Une des quatre images à gauche’

Et le client peut voire les anciennes assurances après l’authentification par ‘username’ et ‘password‘ .

**Mobile modèle** :

Pourquoi les programmes dans les sociétés doit rester rigide et avec deux dimensions ?

Cette modèle encourage à faire des applications avec trois dimensions et des nouvelles méthodes.

Le page principal travail par ‘drag and drop’,

’Drop’ se fait dans un panel de trois dimension qui va élargir la photo choisit pour devient un panel convenable pour l’assurance.

En plus les graphiques qui s’exécutent par un ‘executorservice’, graphique comme le ‘frame’ qui devient plus petit et un autre ‘frame’ travail normalement en même temps.

**Design pattern :**

Deux patterns utilisés dans cette application pour améliorer leur performance et pour faciliter l’utilisation par l’employé.

Première pattern est l’**observer** qui existe dans tous les ‘listeners’, ‘actionlistener’ sur les buttons de ‘login’ et ‘add’, ‘keylistener’ sur le ‘keyboard’ en cliquant ‘ctrl n’ ou ‘ctrl p’. Un de ses deux actions sera utilisé le deuxième pattern qui est le **memento**.

**Memento** facilite le travail de l’employé qui peut remplir les communs ‘textfield’ seulement par un clic. Ce pattern utilise un class qui sauvegarde tous ces informations dans un array list.

**Conclusion**

Ce projet a apporté à nous, une nouvelle forme à approfondir notre connaissance.

Ca nous donne de la compétence pour pouvoir nous adopter à des différents environnements de travail.

D’autre part, le sujet du projet est intéressant.