A web application access for Jobs Observer program and verification

27 décembre 2019 - 22:02

Stetsenko Olexandr
University of Luxembourg
Email : olexandr.stetsenko.001@student.uni.lu

Nicolas Guelfi
University of Luxembourg
Email : nicolas.guelfi@uni.lu

1. Introduction

Le Bachelor Semestre Projet se concentrera sur le développement et le déploiement d'une application web, appelé Jobs Observer, sur un serveur cloud AWS. Le «Jobs Observer » utilises un logiciel existant qui permet de créer des statistiques en se basant sur les emplois dans le domaine du Science Informatique. De plus une recherche dans le domaine du test sera effectuée pour démontrer qu'un programme est correcte.

2. Déscription du projet

Le premier objectif consiste à démontrer qu'un programme est correct. Pour cela deux démonstrations sont développées. La première permet de démontrer qu'un programme est correcte de point de vu de l'exécution. La deuxième démonstration se base sur les comportements prédéfinis pour un programme qui doit les satisfaire.
Le deuxième objectif est de développer une application web qui sera déployé sur un serveur AWS, en utilisant «Django framework » écrit en python. L'application web a pour but de permettre à un utilisateur d'afficher ou de générer une statistique choisie.

3. Production

La production finale du projet est une application web (réf 5.3) et les deux démonstrations (réf 5.1 et 5.2).
La première démonstration permet de démontrer qu'un programme est correcte du point de vu de l'exécution. Tout d'abord, un programme est divisé en multiple bloque qui peuvent être considérées comme des fonctions ou des parties individuelles d'un programme. Après, quelques tests sont définis et réunies dans un ensemble.
Puis, tous les tests dans l'ensemble sont appliqués sur chaque bloque d'un programme. Dans l'ensemble des tests, il y a des tests qui ne peuvent pas ètre appliqués à un bloque. Pour résoudre, ce problème, chaque test est validé avant d'être appliqué. Ensuite, les tests validés sont appliqués sur le bloque. Si le résultat d'un test est le même que le résultat d'un programme, alors le test est réussi.
Si, tous les tests sont réussis alors, le program est correct du point de vu de l'exécution.
La deuxième démonstration utilise le comportement prédéfini pour un programme et la première démonstration. Tout d'abord, les comportements pour chaque bloque du programme sont établies. Chaque comportement est composé de deux fonctions qui permettent d'évaluer les entrées et les sorties d'un programme.
Pour évaluer qu'un comportement est correct, les entrées et les sorties d'un programme et d'un test doivent être dans ce comportement. De plus, le test appliqué doit être réussi. Pour démontrer que tous le programme est correct, tous les comportements doivent être satisfait et tous les tests doivent être réussi.
L'application web développée, est composée de «Front-end » et «Back-end » . La partie «Front-end » est l'interface utilisateur qui permet à un utilisateur d'interagir avec l'application.
L'interface utilisateur est composée de plusieurs pages HTML comme «Home », «About Software », «Plots » et «Statistic ». En utilisant la page de «Plots », l'utlisateur pourra choisir une statistique qui sera affichée sur une page «Statistic ».
La partie du «Back-end » est utilisée pour générer les statistiques à la demande, afficher la partie de «Front-end » et de déployé sur un serveur.

4. Conclusion

Dans ce projet, nous avons démontrer comment un programme peut être correct du point de vu de l'exécution et de comportement en utilisant le raisonnement logique et la déduction naturelle. De plus, nous avons développé une application web qui permet d'afficher et de générer les statistiques.

5. Appendix

5.1. Demonstration of correctness of a program based on unit testing

5.1.1 Assumptions

Let P be a program to test. Assume that the program is syntactically valid.

5.1.2 Demonstrate that a program P is correct for a set of tests

5.1.2.1 Declaration of a program block

$Let\ inputs_{fpb} \equiv \{v \mid v\ is\ a\ value\}$

$Let\ func_{fpb}: inputs_{fpb} \rightarrow output_{fpb}, where\ func_{fpb}\ is\ a\ function$

$Let\ pb_i \equiv (func_{fpb}, inputs_{fpb}, output_{fpb})$

5.1.2.2 Declaration of all program blocks

$Let\ Blocks(P) \equiv \{pb_i \mid pb_i\ is\ a\ program\ block\ of\ P\}$

5.1.2.3 Declaration of a test

$Let\ inputs_{ft} \equiv \{v \mid v\ is\ a\ value\}$

$Let\ func_{ft}: inputs_{ft} \rightarrow output_{ft}, where\ func_{ft}\ is\ a\ function$

$Let\ t_j \equiv (func_{ft}, inputs_{ft}, output_{ft})$

5.1.2.4 Declaration of all test sets for P

$Let\ TA_{pb_i} \equiv [t_j \mid t_j\ is\ a\ test]$

$Let\ TS_P \equiv \cup_{pb_i \in Blocks(P)} TA_{pb_i}$

5.1.2.5 Show the validity and success of a test on a program block

$$success(t_j, pb_i) \equiv \begin{cases} 1, if\ [t_j]_{output_{ft}} = [pb_i]_{output_{fpb}} \\ 0,\ otherwise \end{cases}$$

$$valid(t_j, pb_i) \equiv \begin{cases} 1, if\ [t_j]_{func_{ft}} = [pb_i]_{func_{fpb}}\ and\ [t_j]_{inputs_{ft}} = [pb_i]_{inputs_{fpb}} \\ 0,\ otherwise \end{cases}$$

$$valid(t_j, pb_i) \rightarrow success(t_j, pb_i) = 1$$

5.1.2.6 Demonstrate that the whole program is correct

$$correctBlock(TA_{pb_i}, pb_i) \equiv \forall t_j \in TA_{pb_i} \mid valid(t_j, pb_i)$$

$$correctP\big(TS_P, Blocks(P)\big) \equiv$$

$$\forall pb_i \in Blocks(P), \forall TA_{pb_i} \in TS_P \mid correctBlock(TA_{pb_i}, pb_i)$$

5.2. Demonstration of correctness of a program from requirements point of view.

5.2.1 Considerate the previous parts 1.2.1 to 1.2.4 include as the first part of this proof.

5.2.2 Declaration of requirements

$$Let\ condition\ \equiv\ [v \mid v\ is\ a\ value]$$

$$Let\ conditionsPre \equiv \{condition \mid condition\ is\ a\ set\ of\ values\}$$

$$Let\ conditionsPost \equiv \{condition \mid condition\ is\ a\ set\ of\ values\}$$

$$Let\ pre-condition([t_j]_{inputs_{ft}})\ \equiv$$
$$\begin{cases} 1, \forall input \in [t_j]_{inputs_{ft}}, \exists\ condition\ \in\ conditionsPre \mid input\ \in\ condition \\ 0, otherwise \end{cases}$$

$$Let\ post-condition([t_j]_{output_{ft}})\ \equiv$$
$$\begin{cases} 1, \forall output \in [t_j]_{output_{ft}}, \exists\ condition\ \in\ conditionsPost \mid output\ \in\ condition \\ 0, otherwise \end{cases}$$

$$Let\ r_i \equiv\ (pre-condition([t_j]_{inputs_{ft}}), post-condition([t_j]_{output_{ft}}))$$

$$Let\ RA_{pb_i}\ \equiv\ [r_i \mid r_i\ is\ a\ requirement]$$

$$Let\ RS_P\ \equiv\ \cup_{pb_i \in Blocks(P)} RS_{pb_i}$$

5.2.3 Test satisfaction of a requirement

$$inputsEquality(t_j, pb_i)\ \equiv \begin{cases} 1, if\ [t_j]_{inputs_{ft}} = [pb_i]_{inputs_{fpb}} \\ 0, otherwise \end{cases}$$

$$outputsEquality(t_j, pb_i)\ \equiv \begin{cases} 1, if\ [t_j]_{output_{ft}} = [pb_i]_{output_{fpb}} \\ 0, otherwise \end{cases}$$

$$conditionInput(t_j, pb_i, r_i)\ \equiv \begin{cases} 1, if\ inputsEquality(t_j, pb_i) | [r_i]_{pre-condition([pb_i]_{inputs_{fpb}})} \\ 0, otherwise \end{cases}$$

$$conditionOutput\ (t_j, pb_i, r_i) \equiv \begin{cases} 1, if\ outputsEquality(t_j, pb_i)\ | [r_i]_{post-condition([pb_i]_{output_{fpb}})} \\ 0, otherwise \end{cases}$$

$$satsify(t_j, pb_i, r_i)\ \equiv$$
$$\begin{cases} 1, if\ conditionInput(t_j, pb_i, r_i)\ and\ conditionOutput\ (t_j, pb_i, r_i)\ and\ valid(t_j, pb_i) \\ 0, otherwise \end{cases}$$

5.2.4 A specific requirement is satisfied for a program block

$$Let\ satisfactionR(TS_{pb_i}, pb_i, r_i)\ \equiv$$
$$\begin{cases} 1, if\ \forall t_j \in\ TS_{pb_i} | satsify(t_j, pb_i, r_i)\ \rightarrow\ (\exists t_j \in\ TS_{pb_i} | satsify(t_j, pb_i, r_i)\ =\ 1) \\ 0, otherwise \end{cases}$$

5.2.5 All requirements are satisfied for a program block

$$Let\ satisfactionSetR(TS_{pb_i}, pb_i, RS_{pb_i})\ \equiv$$
$$\begin{cases} 1, if\ \forall t_j \in\ TS_{pb_i}, \forall r_i \in RS_{pb_i} | satisfactionR(TS_{pb_i}, pb_i, r_i)\ =\ 1 \\ 0, otherwise \end{cases}$$

5.2.6 All requirements are satisfied for a program

$$Let\ satisfactionALL\ \equiv \forall pb_i \in\ Blocks(P), \forall TS_{pb_i} \in TS_P, RS_{pb_i} \in$$
$$RS_P \mid satisfactionSetR(TS_{pb_i}, pb_i, RS_{pb_i})\ =\ 1$$

6.3 Application web Front-end

L'application web «Jobs Observer » peut être trouvé sur : http ://ec2-18-215-248-225.compute-1.amazonaws.com/

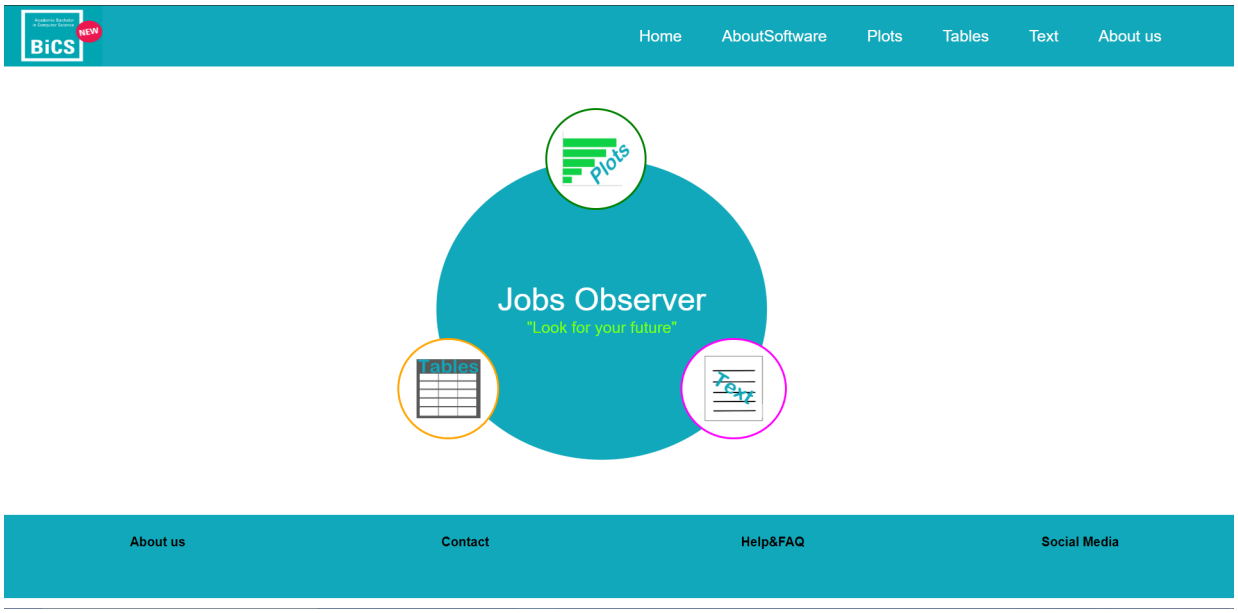

FIGURE 1. La page «Home » du Jobs Observer

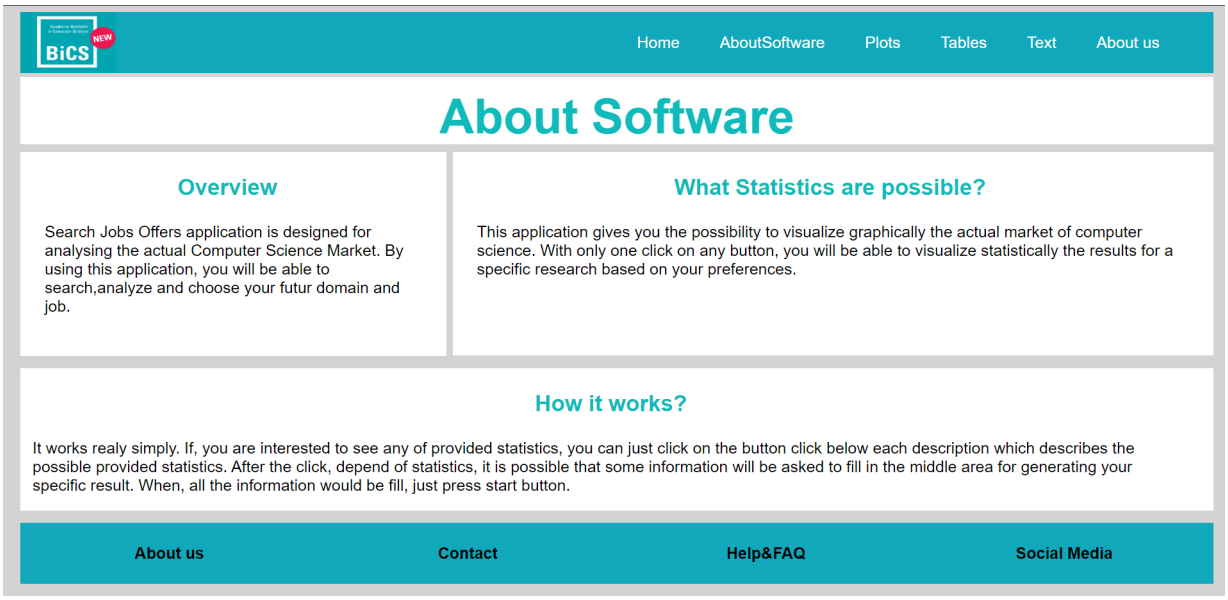

FIGURE 2. La page «About Software » du Jobs Observer

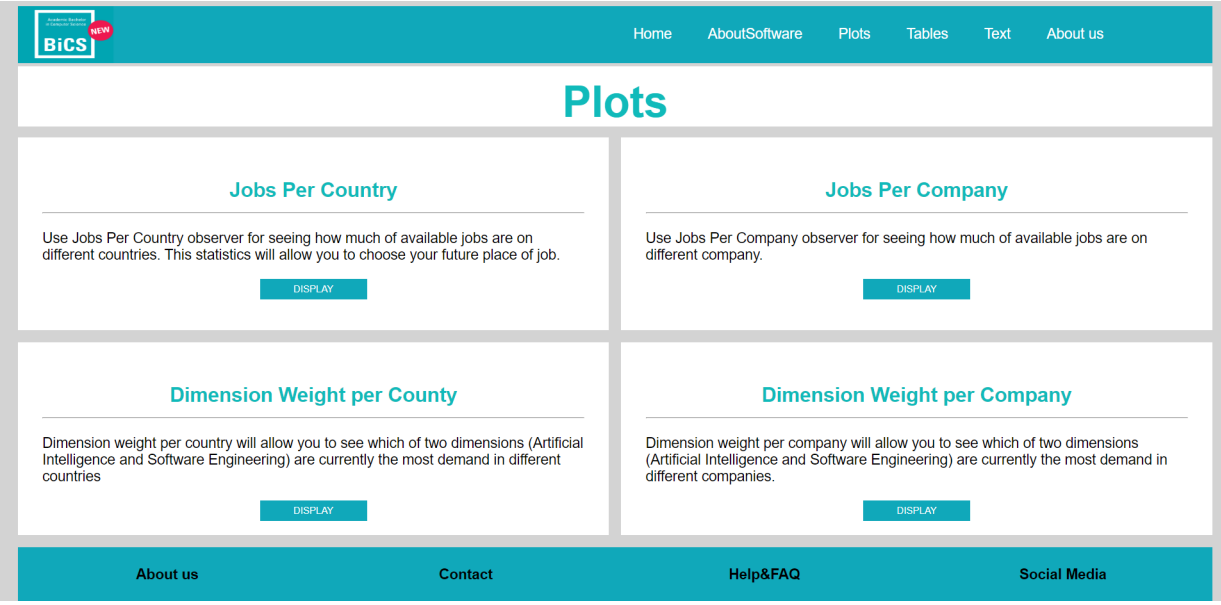

FIGURE 3. La page «Plots » du Jobs Observer

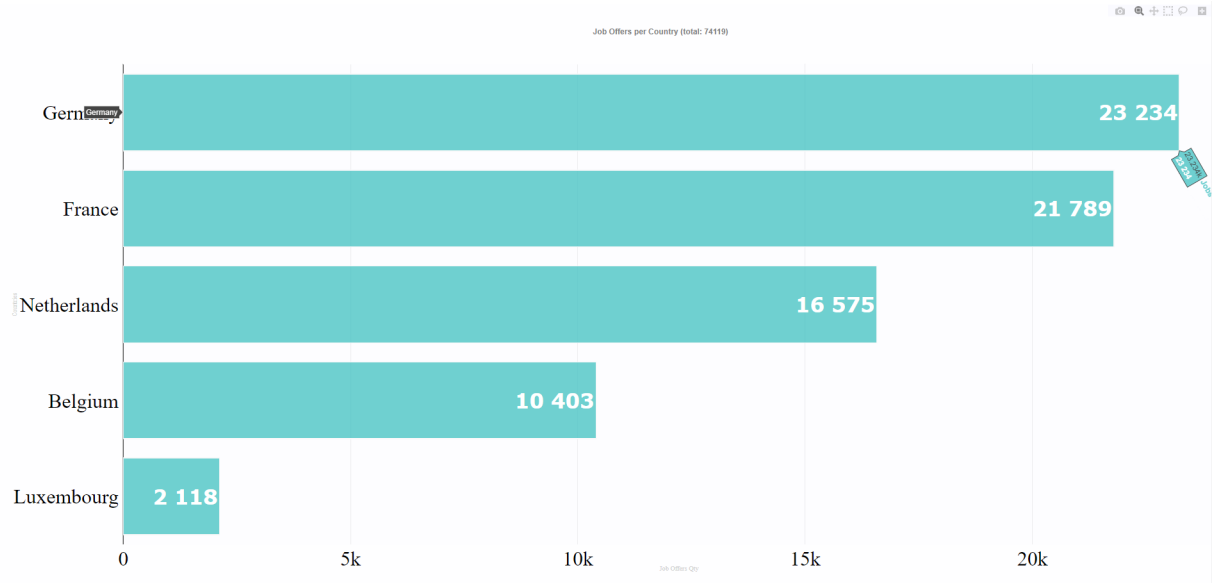

FIGURE 4. Une statistique basée sur les emplois par pays