Bachelor Semester Project - Web application for Computer Science Market Analysis

Monday 9th December, 2019 - 01:33

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

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

***Abstract*—Abstract**

1. Introduction (± 5% of total words)

Computer Science is one of the most important domains in the world and become indispensable in everyday life. As we live in a digital age, there are a lot of industries and companies which focus on improving and exploring new capabilities in this domain. Thus, this led to the grow of complexity and creation of the new sub-fields which request new skills from employees and new places in the companies and industries.

So, a lot of new students who started to study the field of Computer Science may ask themselves, what are the possible fields of Computer Science and what are the possibilities in terms of career perspectives and the availability of jobs per field.For answering theses questions, a software was been developed for performing the actual computer science market analysis. It consists to fetch all available jobs from different countries and companies which belong to the Computer Science domain. Then, based on theses founded jobs, a lot of different statistics can be created such as tables and plots which can be used for observing the trends of the domain. Therefore, by using and observing theses statistics, students will have the possibility to choose the right field and seeing the perspectives for future career.

The Bachelor Semester Project will focus on deploying this recent developed software, called Jobs Observer, on a cloud server in order to make his futures available to all students, who would like to observe the actual Computer Science possibilities. During this project, a small but essential part of the software will be deployed and will be considerate as the basis for future upgrades. Moreover, this part must be operational and include some of possible statistics of the existing software. Then, statistics delivered by the software will be discovered for understanding at best the results for correct implementation. As the software deployment and development are in the context of software engineering, a small research in the field of software testing will be perform. This research will consist to prove the correctness of a program by looking from the scientific point of view.

2. Project description (± 10% of total words)

2.1. Domains

The Bachelor Semester Project is done in different domains of computer science such as Software Engineering, Web development and Software testing.

2.1.1. Scientific domain

Software testing is a process which consists to verify and validate that a software is a bug free. The purpose of software testing is to find errors and bugs in the functionalities of a software that can lead from smaller failures to the malfunction of the overall system. Software testing is divided in verification and validation. Verification[1] process allows to understand, if the development of the software is right. It consists to verify the specified requirements in order to understand if the development of the software is conformed to these requirements. Validation[1] is a process which consists to understand, if the developed software is the right software. This process happened after the development of a software and allow to answer, if the developed software meets the requirements and the needs of the customers.

2.1.2. Technical domains

Software Engineering is based on the principles, which are used in the field of engineering. In Software Engineering, developers develop different types of software which can be small, large, complex or easiest. During the development, they deal with the requirements specification, software design, testing and maintenance. Moreover, developers must apply a structured and disciplined approach for being able to navigate codes, during the development and maintenance. In Software Engineering, developers must base on the customer's requirements, for providing a software that meets their needs. In addition, some other different factors have to be considerate as budget, time constraint or efficiency.

Web development consists to create web applications that are developed, deployed and accessed on the Internet. Web development is divided in front-end and back-end development. Front-end development consists to design the graphical interface of a web application that will allow the interaction between the system and the end users. Usually,

the front-end is composed of several elements that are designed by CSS, HTML or JavaScript. Back-end development is hidden part of a web application. Back-end is composed of three parts such as functionalities of software, databases and server which allows to access and perform operations on the system.

2.2. Targeted Deliverables

2.2.1. Scientific deliverables

The scientific deliverable is a presentation of program correctness demonstration using logical reasoning. The demonstration is based on natural deduction that allows to derived a conclusion based on the premises. For demonstrating the correctness of program, the demonstration is based on the principle of testing that shows if a program is correctly executed or not. Additionally, the demonstration doesn't only show at which conditions the program is considerate correct but why it is correct, based on idea of requirements.

2.2.2. Technical deliverables

The technical deliverable is an operational software deployed on an AWS server. This technical deliverable can be divided into two parts, front-end and back-end. Front-end deliverable is divided into two parts. First part is a mockup of the user interface that is used for the interaction between the system and the end-users. This user interface is easy to understand, has a structure and the basic elements of a webpage. Second part is a user interface based on the mockup that is developed by using CSS and HTML languages and which is compatible with the functionalities from the back-end. Back-end deliverable consists to deploy partially Jobs Observer software to an AWS server. This software is developed by using Django which allows the creation of web applications. The deployed software doesn't include all the functionalities of Jobs Observer but a part of functionalities that are operational and can be accessed through the user interface. The purpose of functionalities is to provide users with statistics based on actual Computer Science Market analysis.

3. Pre-requisites ($[5\%..10\%]$ of total words)

3.1. Scientific pre-requisites

3.1.1. Proof

Proof is a logical demonstration which shows the truth of a proposition based on assumptions or axioms. For writing a proof, there are no only one method which can be used but several methods s.t direct proof, mathematical proof, natural deduction proof and more. During this project, a proof is developed based on natural deduction.

3.1.2. Natural deduction

Natural deduction[2] is an intuitional method which allows to develop a proof. In Natural deduction all the developed proof is considerate as the proof by hypothesis. The proof is developed based on smaller proofs which are then combine at the end based on the rules for providing a justification to an assumption or hypothesis. Usually a proof is written as sequence of lines where for each line the justification is based on the any previous line, also called premises.

3.1.3. Unit testing

Unit testing is a way of testing which purpose is to test the separate parts of a program and shows that each individual part is working correctly. For showing the correctness of execution, some requirements are defined which include the inputs and the correct outputs. In unit testing, the developer knows in advance what is the block of code to test. So, based on these knowledges, a test code will be produced for determine if a block of code to test behaves as expected. For showing that a code behaves as expected, the block to test will accept some predefined inputs from the requirements and will produce an output. Then, the test code will check if the output of the tested code is correct with respect with the output from the requirements.

3.2. Technical pre-requisites

3.2.1. Python[2]

Python is an interpreted, object-oriented, high-level and multiplatform programming language dedicated for web and application development. It has a dynamic semantics [3] which define how and when the various constructs of a language should produce a program behavior. Python programming language allows a dynamic typing which gives the possibility to change type of value of a variable to another type without receiving an error. Additionally, it has a dynamic binding [4] option which allows to a computer program to wait until runtime for binding the name of a method for the actual subroutine. The reason for using Python programming language in this project, is that it is easy to learn, to maintain and to develop. Moreover, it offers a possibility to divide a program in modules which allow to have a structure and improve organization. Additionally, some different packages can be imported from python library and they are used for facilitate the program development. The advantage to use modules and packages, is that they can be imported anywhere in the program and their functionalities can be easily reused.

4. A Scientific Deliverable 1

For each scientific deliverable targeted in section 2.2 provide a full section with all the subsections described below.

4.1. Requirements ($\pm$ 15% of section's words)

Describe here all the properties that characterize the deliverables you produced. It should describe, for each main

deliverable, what are the expected functional and non functional properties of the deliverables, who are the actors exploiting the deliverables. It is expected that you have at least one scientific deliverable (e.g. "Scientific presentation of the Python programming language", "State of the art on quality models for human computer interaction", ...) and one technical deliverable (e.g. "BSProSoft - A python/django web-site for IT job offers retrieval and analysis", ...).

4.2. Design (± 30% of section's words)

Provide the necessary and most useful explanations on how those deliverables have been produced.

4.3. Production (± 40% of section's words)

Provide descriptions of the deliverables concrete production. It must present part of the deliverable (e.g. source code extracts, scientific work extracts, ...) to illustrate and explain its actual production.

4.4. Assessment (± 15% of section's words)

Provide any objective elements to assess that your deliverables do or do not satisfy the requirements described above.

5. A Technical Deliverable 1

For each technical deliverable targeted in section 2.2 provide a full section with all the subsections described below. The cumulative volume of all deliverable sections represents 75% of the paper's volume in words. Volumes below are indicated relative the the section.

5.1. Requirements (± 15% of section's words)

In this Bachelor Semester Project, there are several requirements that must be satisfied. These requirements are divided into two parts: front-end and back-end. For each part, functional and non-functional requirements will be introduced.

5.1.1. Front-end

5.1.1.1. Front-end functional requirements

In this section, we will see the most important functionalities that the front-end must satisfies. The front-end consists to design a GUI that allow the interaction between web-server and end-users.

FR1: Graphical User Interface should be created by using static pages in order to obtain a first visual view of the web application. These static pages must have some basic elements of a webpage that will allow simple interactions. Moreover, some buttons must be created for displaying different statistics based on jobs offers.

FR2: The GUI has to be connected to the back-end functionalities. This connection should be implemented by using the GUI buttons which will be connected to the python script for displaying a statistic on a new static page.

5.2. Front-end non-functional requirements

NFR1: Friendly interface. The GUI should be easy to use without spending a lot of time for learning. Moreover, the user interface has to be intuitive so that the user can expect himself in advance the result produced by clicking on an element of user interface. Moreover, the basic elements of a webpage must be found in the right places.

NFR2: Readability: The font-size of components of GUI must be easily visualize so that the meaning and the information transmit of an GUI element can be easily understand by the user. Moreover, each static page must be not overload with the data and textual information because a lot of data can affect the user understandability.

5.3. Back-end

5.4. Back-end functional requirements

In this section, we will see the most important functionalities that the back-end must satisfies. The back-end part consists to develop a web-server that allows the creation of statistics and is deployed on an AWS server.

FR1: The web-sever development must be realized by using Django framework which is written in Python and the compatibility with between Django web-server and AWS server has to be ensured.

FR2: The old database from the existed software has to be used in the Jobs Observer. As the database file is to big, it should be imported from an external source such Dropbox where it will be downloaded and used. As, the old database is designed by using peewee library, which is a local database, Jobs Observer has to reuse the codes from the existed software for making the database working.

FR3: Operations. All the operations for creations of statistics have to be imported to the Jobs Observer in order to apply them on GUI buttons. When a statistic is requested by clicking on front-end button, some of operations have to be used in the right order for producing a statistic. Moreover, the operations to use must be operational and provide the tight results. Additionally, the creation of all statistics must be made in a particular place of the web-sever.

FR4: The back-end must be connected to the front-end in order to execute the operations when they are requested at front-end and receive a visual representation of results.

5.5. Back-end non-functional requirements

NFR1: Maintainability. The development of web-server must be done in a way that it can be maintainable after the end of the project. For ensuring this requirement, a structural approach has to be used, some comments have to be written for the understandability of the program. The right naming must be used for operations and variables.

NFR2: Operationality. The web-sever must be operational after the development for allowing the execution of different statistics and to be able to provide some firsts results.

NFR3: Compatibility. The created web-server must be compatible with AWS for allowing Jobs Observer to run continuously and non-locally.

5.6. Design ($\pm$ 30% of section's words)

5.7. Project creation

The first step for designing the project, is the creation of Jobs Observer on AWS. The creation is realized by using a development tool called CodeStar which allows the creation of simple project and also more complex project such as a web-server. For creating the project, a Django framework is used, based on the Python language. Additionally, the project source code of the project can be found on github which allows to perform an incremental development.

5.8. Graphical User Interface

Before passing to the development part of web application, we will focus on the Graphical User Interface design. The first part consists to create a mockup by using an external online tool for creation of web sites. The creation of mockup is started by defining the structure of each page such as the locations of buttons, images and content. Then, the order in which these pages can be accessed was been imposed in order to ensure the logical passage between one page to another. There are four pages that are designed such home, about software, plots and statistic which can be considerate as the basis for the future development. Then, the designed mockup was been used for created HTML pages that represents the content and structure and which is support by CSS files that is responsible for design of each HTML page.

5.9. Django structure

Django framework has a structure which allow the creation of web application. This structure is divided into two folders which allows the development and configuration of a web application. The first folder, called "ec2django" is used for the configuration. The second folder, called "helloworld" is used for the development of application.

5.10. Configuration

The "ec2django" folder is used for the configuration and contains four python modules such init, settings, urls and wsgi. The settings.py module is the most important and is the only one module used in this folder. It allows to setup of application parameters such as path to HTML static pages, type and location of database and the middleware. During the development, this module was been used for indicating the location of CSS files that are reused by HTML files for design. Moreover, it is used for performing some changes in middleware classes for allowing local tests.

5.11. Development

The "helloworld" folder is used for the development of a web application (Notice: the "helloworld" name is the base name of application). As previously, this folder contains some modules such as init, admin, apps, models, views and urls. For the explanations, we will focus only on the last three modules.

5.12. models.py

This module is use for creation of database and declaration of database structure. As, we have already an existed database from the existed software, this old database is reused for fetched the available jobs. Moreover, the oldest database was been created by using peewee package and the structure of database were been previously defined. So, the code for existed structure is reused for interaction with the database and retrieve the data. Unfortunately, the existed database is too big (around 1.2GB) and can not be used directly from the repository. So, this database is manually stored into a cloud server, called dropbox. When this module is executed, this stored database file is download to an empty database file.

5.13. views.py

This module is used for displaying different pages and allowing the interaction between the GUI and users. This module is composed of several functions, called "views", which accept a request and return a HTML page based on this request. As, we have design previously different HTML pages for our web application, these functions are used for displaying the design HTML pages. Additionally, it is possible to perform some additional operations that allow the creation of different statistics. When, a user will call for an action, a request will be sent to the system and base on this request a certain operation can be performed and a HTML page is displayed. Moreover, this module is used an imported module, called "stats.py".

5.14. urls.py

This module is included in the Django structure. It is used for indicating different urls which are requested when

a view function is requested in views.py module. So, if a function is requested and return a HTML page, this HTML page will have a previously defined url.

5.15. stats.py

This module is one of the most important modules because it is used for creating different statistics based on available jobs in Computer Science. This module contains many functions that can be used for creating statistics. Usually for creating a statistic, a dataframe is firstly created and base on the created dataframe, a plot or a table is created which represent a statistic. For creating the statistics, this module uses previously imported database for retrieving the data.

5.16. Production (± 40% of section's words)

In this section, the description of the deliverable concrete production will be explained. For this section, the deliverable is divided into two parts, front-end and back-end deliverable. Notice that this deliverable is only a prototype and represent the basis for future development.

5.16.1. Front-end deliverable

Front-end deliverable is a Graphical User Interface view which is composed of several HTML pages that are used for the interaction with the web-application. In total the front-end deliverable contains four HTML pages such as "Home", "About Software","Plots" and "Statistic".

5.16.2. Home page

The "Home" page is the first HTML page which show to the user when he accesses the web application. The purpose of this page is to show what kind of statistics are offered by the web application and to redirect a user to other pages where he will be able to find the descriptions of possible statistics or some general information about the software.
The "Home" page is separate into three parts such top, middle and down. The top of the page is a menu bar which is composed on a "BiCS" image on the left and some buttons to the right. There are six buttons to the right such "Home", "About Software", "Plots","Tables","Text" and "About us" which are use for redirect the user to other HTML pages. At the moment, there are only three first buttons that are operational.
The middle of the page is composed of four circles which are the clickable buttons. The biggest button contains the name and a logo and is used for redirect the user to an "About Software" page. The other three small buttons include images which show what can of statistics are possible and based on the title inside of images redirect the user to the right pages. The down of the page is reserved for additional information such as contact, Help&FAQ and Social Media but not yet completely finished.
For the next pages is top and down of each page is the same as in the "Home" page.

5.16.3. About Software page

The purpose of this page is to provide a user with the general information about the software. This page provides three types of information. The first is an overview about the software where the purpose of web application is described. The second describe what are the possible statistics and the third describe how the software works. Each description is place into a white box label.

5.16.4. Plots page

The purpose of this page is to provide a description what kind of statistics are possible and to allow a user to choose a statistic to display. The description of each statistics can be found in a white box composed of a title (the name of statistic) followed by a small text of expected result. At the bottom of each white box, there are a button "DISPLAY" which is used for displaying a statistic on another page.
On this page, there are right now four description of statistics among which two are operational, "Jobs Per Country" and "Jobs Per Company". These four descriptions are structured as follow, there are two statistic description per row and two per column.

5.16.5. Statistic page

This page appears when a user clicks on a "DISPLAY" button from the "Plots" page and show to user an interactive statistic. At the moment, only the statistic is displayed without any additional information about it. From the mockup point of view, the idea of this page is to display a window for the statistic and to add an additional information around.

5.16.6. Back-end deliverable

Back-end deliverable is a web application deployed on an AWS server. The web application is developed in Django and is composed of several modules. In this section, the concrete implementation of modules will be described such models.py, views.py, urls.py and stats.py and their relations.

5.16.7. models.py module

This module is used for the database creation and declaration of database structure. At beginning, an existed database file, called "jobs.db" is downloaded to the system from the Dropbox by using dropbox python package. This download is possible only, if it exists inside the system another database file. So, the "jobs.db" database file from Dropbox is downloaded into an empty database file also called "jobs.db" and this new file will be used as the new database. The reason to perform this download operation is that the existed database file from the previous software is too big and can't be store on github after that the project is already modified from his initial state.
Then, a structure for the database is declared for accessing the data and different tables of database. The database is composed of three types of table such Job, SearchQuery and Token. Job table is composed of several attributes such id, county, company, etc. SearchQuery table is composed of a term and a place which indicate the location. Token is composed of jobs, values and order which is used for indicating the number of available jobs.

5.16.8. stats.py module

This module can be found in "modules" folder. This module is completely imported from the existed software and is used for creating different types of statistics. It provides three types of statistics: tables, plots and text which should be only created as HTML pages. In this module some changes are performed compare to initial version in order to allow the correct execution on the server. The first change consists to change all path to more common paths. Another, change consists to remove the read Excel sheet with all keywords because AWS server doesn't allow to load the Excel sheet as a dataframe locally and externally. There exists a solution for this problem which can be to create a new database externally and load the data from the Excel sheet into it.

5.16.9. views.py module

This module is used for interacting between the front-end and back-end and it can be accessed from the "helloworld" folder. It is composed of different functions called "views" that accept a request and return a HTTP response to an HTML page. In this module there are three functions: get, change_view and statistics_view.
The get function is used for showing the "Home" page when a user accesses the Jobs Observer. The change_view function is used for returning a HTTP response for "About Software" and "Plots" page. This function is executed when the request method is GET request. This GET request is represent as a dictionary with one value inside which indicates the name of the button in the HTML page. So, for this function the names are "aboutSoftware" and "plots" which represent the names of buttons in HTML.
The statistics_view function is used for displaying statistics requested by the user from plots page. The principle of execution is the same as for the previous function but instead directly to return a HTTP response for an HTML page, some operations have to be performed in between for creating a statistic HTML page. At the end a HTTP response for the creating HTML page is returned. The operations performed are the operation from the stats.py module.
Let's take an example. A user request for a statistic, called "Jobs Per Country". This request will be sent to views.py module and will pass this request through all the functions. The value of GET request is "displayPlot1" and if this value is true for a certain "if" method such if request.method == "GET" and "displayPlot1" in request.GET then some operations from stats.py will create a statistic which is used for returning a HTTP response.

5.16.10. urls.py

This module is the last module before displaying a HTML page. It is composed of a list of urls where each name of url is associated to a function from views.py module. As example, the url called "AboutSoftware" is associated to change_view function. Notice that the url "AboutSoftware" is not a full url but a part of url which is considerate as the continuation of the basic url s.t basic url + "AboutSoftware".

5.16.11. Project on AWS

Jobs Observer web-application is already deployed on AWS. The deployment is realized by putting the project on github from where the AWS take the source code and performed a build and deployment itself.

5.17. Assessment

In this section, the satisfaction of technical deliverable with respect to the requirements are compared and evaluated. As previously, the front-end deliverable and back-end deliverable evaluated based on requirements.

5.17.1. Front-end deliverable

The front-end deliverable which is the GUI interface of our web application satisfies all the functional requirements. The accomplishment of the first requirement can be seen on the figure(number) which shows that the GUI contains some basic elements of a web page. Additionally, the created page is a static page and contains some buttons that allow the creation of statistics.
The second requirement is also satisfied because the buttons for displaying statistics are connected to the python script and are operational.
The front-end deliverable also satisfies all the non-functional requirements which consist to have a friendly interface and a good readability. For the first requirement, a simple interface is created for avoiding to user to spend a lot of time for learning because based on the names of buttons and titles, he would be able to predict the result in advance.
For the second requirement, the GUI interface doesn't overload the user with a lot of information which increase the readability. Additionally, the font-size is adequate because it allows to see all the names of elements and to read all the text sections.

5.18. Assessment ($\pm$ 15% of section's words)

cf. section 5 applied to the technical deliverable

Acknowledgment

The authors would like to thank the BiCS management and education team for the amazing work done.

6. Conclusion

The conclusion goes here.

References

[1] Verification vs Validation." Software Testing Fundamentals, 3 Mar. 2018. http://softwaretestingfundamentals.com/verification-vs-validation/

[2] "3. Natural Deduction for Propositional Logic." 3. Natural Deduction for Propositional Logic - Logic and Proof 0.1 Documentation https://$leanprover.github.io/logic_and_proof/natural_deduction_for_propositional$

[3] "What is Django?" Python For Beginners https://www.pythonforbeginners.com/learn-python/what-is-django/

[4] "Fr." Amazon, Sundance Pub., 2002 https://aws.amazon.com/fr/what-is-cloud-computing/

[5] Miller, Cody. "What Is Cloud Computing & How Does 'The Cloud' Work?" Fastmetrics Business Blog, 14 Nov. 2019 https://www.fastmetrics.com/blog/tech/what-is-cloud-computing/

7. Appendix

All images and additional material go there.