POLYLAB: A CUSTOMIZED MULTI-LANGUAGE DEVELOPMENT ENVIRONMENT FOR COMPUTER SCIENCE STUDENTS IN NIGERIAN POLYTECHNICS

, Onuora, Augustine Chidiebere[1], Madubuike, Chibuike Ezeocha[2]. & Ezeorah, Ezekiel Uchechi[3]

[1,2,3] *Computer Science Department, Akanu Ibiam Federal Polytechnic Unwana Afikpo, Ebonyi State, Nigeria*

---***---

Abstract - *In this paper, a customized multi-language development environment (MLDE), PolyLAB, is developed for students in Nigerian Polytechnics. The online programming laboratory supports the entire Nine (9) programming languages offered by computer science students in Nigeria Polytechnic at both National Diploma (ND) and Higher National Diploma (HND); namely Java, Python, C, C++, MATLAB, PASCAL, HTML, CSS and JavaScript. Using the content management system of PolyLAB, Lecturers and Technologist in Nigeria Polytechnics can post lesson materials on programming courses taught by them for their students to study while the integrated multi-language code editor will allow the student try program codes on the fly. Successful implementation of PolyLAB will remove the burden of both the technicalities and the cost of both setting up and maintaining different development environment used by Computer Science student in Nigerian Polytechnics, as well as facilitate independent studies from individual comfort locations.*

Key Words: Computer Science (CS), Multi-language development environment (MLDE), Nigerian Polytechnics, Online Programming Laboratory (PL).

1. INTRODUCTION

A Multi-language development environment (MLDE) is an integrated development environment (IDE) that is designed to support multiple programming languages allowing developers to work with a variety of languages in a single environment [2]. Some of these MLDE can be in mobile, desktop or cloud/web version. Presently the cloud version of MLDE enjoys more preferences from developers globally because of its support for remote access, automatic backup and collaborative coding.

The National Board for Technical Education (NBTE) established in 1977, is the principal organ of the Nigerian Federal Ministry of Education which saddled with the task of supervising, regulating and overseeing educational programmes offered in Nigeria Polytechnics, monotechnics, and technical colleges/secondary schools. As part of its regulatory role, NBTE reviews their curriculum every five (5) with the latest curriculum which was reviewed in 2019. Students studying computer science are expected to learn at least eight (8) different programming language at both ND and HND level before graduation [7].

Though the reasons for multi-language training as could be seen in the case of a Computer Science programme in Nigerian Polytechnics are enormous; ranging primarily from: the need to expose the young programmer to different programming paradigms, practices and theories; to the concepts of polyglot programming and micro-services adoption [8]. While polyglot programming allows a programmer to learn as many programming languages as possible in other to position him/herself for better job, more money and flexible choice of tools in software career; micro-services concept allows different independent applications to be coupled into one project so that the project can enjoy the benefits each programming language used. An example is the case of combining Java and C applications into one project in other to enjoy both the portability of Java and the resource management of C language. According to [9] some of the contemporary application developers use more than one programming language to develop their applications. Examples Magento and X-cost were developed with over ten (10) programming language.

However, the financial and technical burden of setting up and maintaining both the hardware and the software dependencies of such multi-language training laboratories of Nigerian Polytechnics and individual systems of the students involved cannot be denied. Also, while some existing online multi-language development environment (MLDEs) like CodingHere and SoloLearn, are generic in the choice of programming language they support others like GitHub and Google Colab will still subject the developers to setup the environment by themselves [5]. There is need to deploy a customized multi-language development environment (MLDE) that will specifically handle the identified challenges of a multi-language training systems of the Nigeria Polytechnics.

In this work, a multi-language development environment was developed to remove the burden of both the technicalities and the cost of both setting up and maintaining different development environment associated with studying Computer Science in Nigerian Polytechnics. The new system, PolyLAB, allows students in Nigeria polytechnics to write and run programs in any of the nine (9) programming language from a single interface as well as allow lecturers and technologist to provide study materials to their students using its content management system.

2. RELATED WORKS

[3] developed a multi-language execution platform using a flexible virtual virtual machine (VVM). Unlike the rigid traditional virtual machines (VM) which is bytecode-specific, VVM can execute programs written in different bytecode language depending on its configuration. To achieve its flexibility, VVM creates separate virtual machines for each programming language called (VMlet). Each VMlets contains a virtual machine model, a memory model and an environment model configured for a particular bytecode language. So whenever a particular bytecode language is to be executed, the VVM will load the appropriate VMlet for it. The transparency in loading and offloading of the VMlets makes VVM unique.

Similarly, [6] proposes the implementation of a single live programming environment (LPE) that will support different programming code editors or different integrated development environments (IDEs). The Poly2Kanon project is an extension of Kanon, a live data structure programming environment for JavaScript. Using their proposed system, a programmer will not need to separately configure live servers for each development environment used for developing multi-language systems. For each adopted language, Poly2Kanon separates the language code editor from the language runtime; then uses a live server to join/bridge the two. Communications between the components are done using APIs.

As a solution to the existing paid, legacy language source code analyzers and visualization tools, [10] developed Analizo. Analizo was developed as a free multi-language and extensible source code analyzer and visualization tool. Analizo supports C, C++, and Java. Unlike some of the existing legacy code analyzers like CScope, LDXC and CTAGX with outdated libraries, due to the absence of developers' support, Analizo provides interfaces for adding new features so that it will continue to enjoy both developers and researchers support. With Analizo, developers and researchers can both identify and fix potential code problems as well as optimize their codes using the result of the analysis.

Using PHP, [1] developed a cloud-based multi-language development environment. Cloud MLDE is deployed on a cloud provider, Windows Azure, and supports four (4) programming language namely: Python, Java, C and C++. However, the system does not integrate web API, mobile version and collaborative editing.

Due to the absence of the integration of cross-language relations, most existing integrated development environment (IDE) are not designed to directly support the development of multi-language system (MLS). [9] developed two different multi-language development environment prototypes, TexMo, and Coral. The two systems will provide visualization, navigating, static checking, and refactoring as its cross-language relation integration.

3. DESIGN METHODOLOGY

Agile methodology is the software development methodology adopted in this project. As one of the most common software development approaches today, agile methodology adopts a different strategy than the traditional linear approach by focusing on how to satisfy the consumers instead of emphasizing on documentation and strict procedures. In agile, jobs are divided into manageable sprints that take one to four weeks to finish. Agile software development's main advantage is that it enables iterative software releases. Iterative releases increase productivity by enabling teams to identify and correct flaws and set expectations early on. With regular incremental enhancements, they also enable consumers to enjoy the benefits of software earlier.

With agile, the user requirements and system requirements were extracted and elucidated as functional requirements and non-functional requirements. These functional requirements assist one in capturing the intended behavior of the proposed system. Such behavior may be represented in terms of actions, processes, or tasks that a system must carry out. The identified functional requirement for the proposed system are:

1. The software authenticated users (Staff and Students) with their staff ID or student Matric Number.
2. Only users duly approved can use the system.
3. The background color for the Code Editor windows in the application will be black and have a hexadecimal RGB color value of 0x000000. This will help reduce the effect of light emission to the eye of students or staff using the editor.
4. Only Managerial level employees have the right to view and inspect other staff and students.
5. The software system should be integrated with VSCode API
6. A Verification email is sent to user whenever he/she registers for the first time and is approved to use the software system.

While the non-functional requirements include:

1. Emails should be sent with a latency of no greater than 12 hours from such an activity.
2. The processing of each request should be done within 10 – 20 seconds
3. The Code IDE should load in 5 seconds when the number of simultaneous users are > 1000

4. The Code IDE can be used on any portable devices like mobile phones (Android or IPhone), IPad, tablets.

5. The software should be portable. So moving from one OS to other OS does not create any problem.

6. Users must change the initially assigned login password immediately after the first successful login. Moreover, the initial should never be reused.

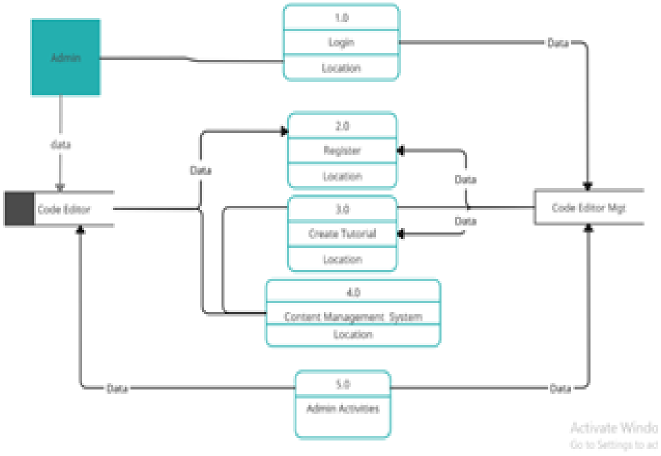
Some selected technologies were used in the development of a system in other to meet the functional and non-functional requirements. HTML and CSS helped in the designed forms for user inputs while PHP was used to access these inputs and send them to the database. PHP was also used to retrieve users' login details during authentication and the display of profile from data stored on the MYSQL DBMS. For the non-functional requirements, the use of CSS in the form of POSTCSS and TailwindCSS made the application to be responsive and thereby portable. This made it to be able to render perfectly on various computing devices. The use of HTML, CSS and JavaScript made it be a cross-platform application, running on the browser of any OS such as windows, Mac or Linux efficiently.

More so, many factors that were considered while designing the proposed system. Some of these design considerations include the ability of the end-user to easily interact with the user interface (usability); the ability to add new capabilities to the software system without making significant changes to the architecture (extensibility); the ability to co-exist and cooperate with existing software systems (compatibility); the ability to exist as a collection of testable and maintainable components (modularity); and the ability to easily apply bug fixes and modifications (maintainability)..

4. RESULT AND DISCUSSION

The result of the design and the implementation of the new system are presented as data flow diagram, user-case diagram, and snapshots of the user interface.

The data flow diagram shows the different procedures the system data undergoes in other to be processed starting from the point of initial input to final output/storage.



a

Fig.1: The dataflow diagram of the new system (source: author)

The use case diagram presents the high level diagram of the new system. Showing its users and the different action they can perform. From Fig. 2, one can identify that the new system has four (4) users (actors) who can perform six (6) different activities based on their roles.

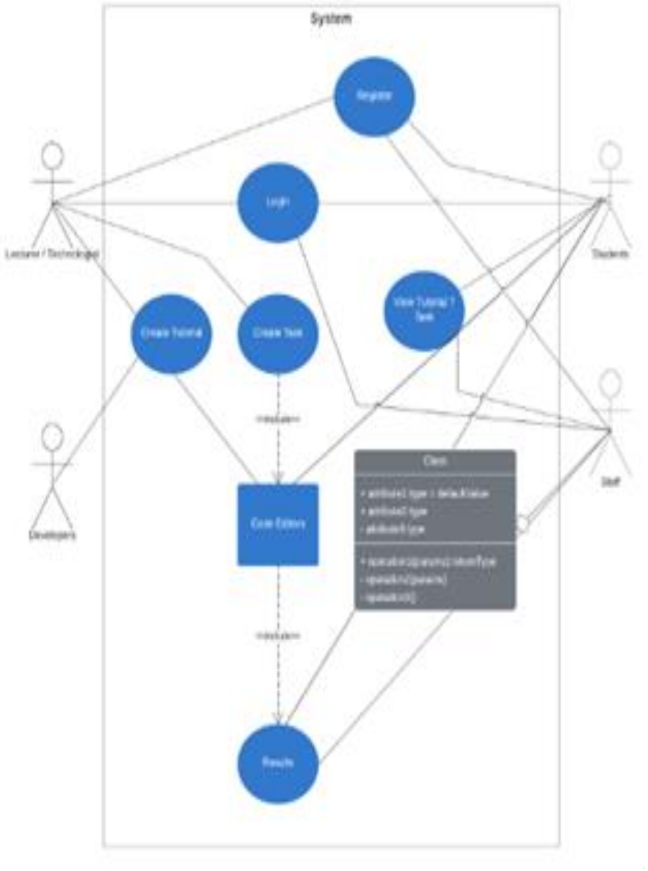
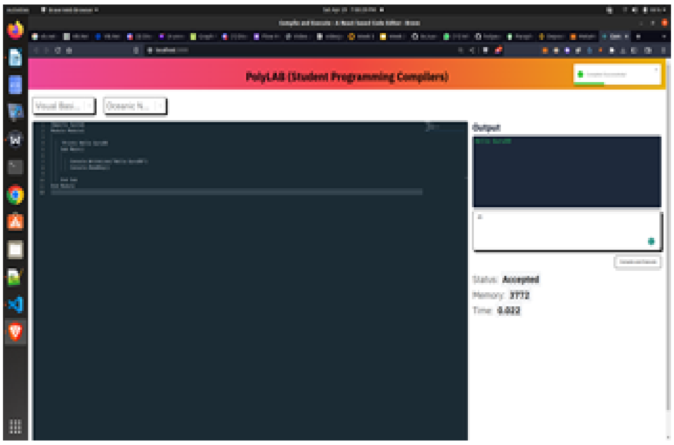


Fig. 2: The use-case diagram of the new system (source: author)

The entire system is divided into two main modules with several sub-modules and components. The two main modules are: (a) The Web Portals: This handles the registration, Login, Lost Password, Tutorial publishing, Task / Assignment publishing. The technology employed here is a Content Management System (CMS). The essence of using a CMS built with HTML, CSS, JavaScript for frontend and PHP/MySQL for backend and database management. (b)The Code Editor: This handles the compilation of programming languages. The editor was built with React, a JavaScript framework and connected with API from RapidAPI. THE tech stack here includes, HTML, TailwindCSS, PostCSS, JavaScript and Firebase. The user interface of PolyLAB is presented in Fig. 3

Successful implementation of PolyLAB will remove the burden of both the technicalities and the cost of both setting up and maintaining different development environment used by Computer Science student in Nigerian Polytechnics, as well as facilitate independent studies from individual comfort locations through the use of internet [11].



Status: Accepted
Memory: 3772
Time: 0.022

Fig. 3: The user interface of the new system (source: author)

5. CONCLUSIONS

In this study, a customized multi-language development environment (MLDE), PolyLAB, is developed for students in Nigerian Polytechnics in order to eliminate the financial and technical burden faced by both students and the Departments in both setting up and maintaining the different programming languages studied by Computer Science Students in Nigeria Polytechnics. To enable a better understanding of the new system, artifacts like the use case diagram, dataflow diagram and system snap shots have been included alongside with the design details. The online programing laboratory is equipped with both a content management system and a code editor to facilitate both publication and the on the fly coding and running of source codes. Further studies will concentrate on provision of web API, mobile version, and integration of code assessment, code version tracking and code plagiarism test.

ACKNOWLEDGEMENT (Optional)

The authors wish to acknowledge Tertiary Education Trust Fund (TETFUND) IBR for their grant toward the actualization and Success of this research.

REFERENCES

[1] Abdulla, S., Iyer, S. & Kutty, S. (2013). Cloud Based Compiler, International Journal of Students Research in Technology and Management, 1(3): 308-322.

[2] Azumo (2023). Integrated Development Environment, accessed from https://azumo.com/insights/integrated-development-environment on March 11, 2013

[3] Folliot, B., Piumarta, I. & Riccardi, F. (1998). A Dynamically Configured Multi-Language Execution platform, In Proc. ACM SIGOPS European Workshop 1998: 175 -181.

[4] Kurniawan, A, Kurniawan, A. Saesanto, C. & Wijaya (2015). CodeR: Real-time Code Editor Application For Collaboative Programming, International Conference on Computer Science and Computational Intelligence (ICCSI 2015), 59(2015), 510-519.

[5] Madubuike, C. E., Onuora, A.C., Ezeorah, E. U. (2023). A Review of Virtual Programming Laboratory: Design Issues, International Research Journal of Engineering and Technology (IRJET), 10(2), 1-6.

[6] Masuthara, H., Takahashi, S., Izawa, Y & Cong, Y. (2020). Toward a Multi-Language and Multi-Environment Framework for Live Programming, Proceedings of Live Programming Workshop 2020 (Live'20) on November, 2020.

[7] NBTE (2019). National Board for Technical Education: Curriculum and Course Specifications, Kaduna, April 2019

[8] Neale, D. (2023). Why You Should Become a Polyglot Developer, accessed from https://cult.honeypot.io/reeds/become-a-polyglot-developer on March 11, 2023.

[9] Pfeiffer, R. & Wasowski, A. (2015). The Design Space of Multi-language Development Environment, Softw. Syst. Model, 14: 383-411.

[10] Terceiro, A., Costa, J., Miranda, J., Meirelles, P., Rios, L. R., Almeida, L., Chavez, C. & Kon, F. (2010). Analizo: An Extensible Multi-language, Source Code Analysis and Visualization Toolkit, In Proceedings of Brazilian Conf. Softw, Theory Practice, Tools Session, Jan. 2010: 107.

[11] Ezeano, A. N.; Idemudia, O. J.; Madubuike, C. E.; Omoregbee, E. U. and Onuora, A. C. (2017). Web Based Programming: A Veritable Tool for security and National Development. International Journal of Computer Applications Technology and Research, Vol. 6, Issue 8, September, 2017, ISSN:2319-8556.