sustainability

Article

Sustainable Project-Based Learning Methodology Adaptable to Technological Advances for Web Programming

Juan Carlos López-Pimentel [1,*], Alejandro Medina-Santiago [2], Miguel Alcaraz-Rivera [1] and Carolina Del-Valle-Soto [1,*]

[1] Facultad de Ingeniería, Universidad Panamericana, Álvaro del Portillo 49, Zapopan 45010, Mexico; malcaraz@up.edu.mx
[2] Coordinación de Ciencias Computacionales, Instituto Nacional de Astrofísica Óptica y Electrónica, Luis Enrique Erro #1, Santa María Tonantzintla, Puebla 72840, Mexico; amedina@inaoep.mx
* Correspondence: clopezp@up.edu.mx (J.C.L.-P.); cvalle@up.edu.mx (C.D.-V.-S.)

Abstract: The fast pace of development of the Internet and the Coronavirus Disease (COVID-19) pandemic have considerably impacted the educative sector, encouraging the constant transformation of the teaching/learning strategies and more in technological areas as Educational Software Engineering. Web programming, a fundamental topic in Software Engineering and Cloud-based applications, deals with various critical challenges in education, such as learning continuous emerging technological tools, plagiarism detection, generating innovative learning environments, among others. Continual change and even more change with the current digitization becomes a challenge for teachers and students who cannot depend on traditional educational methods. The article presents a sustainable teaching/learning methodology for web programming courses in Engineering Education using project-based learning adaptable to the continuous web technological advances. The methodology has been developed and improved during 9 years, 15 groups, and 3 different universities. Our results demonstrate that the methodology is adaptable with new technologies that might arise; it also presents the advantages of avoiding plagiarism in students and a personalized induction for every specific student in the learning process.

Keywords: programming approach; web programming; sustainable education; teaching experience; teaching methodology

check for updates

Citation: López-Pimentel, J.C.; Medina-Santiago, A.; Alcaraz-Rivera, M.; Del-Valle-Soto, C. Sustainable Project-Based Learning Methodology Adaptable to Technological Advances for Web Programming. *Sustainability* **2021**, *13*, 8482. https://doi.org/10.3390/su13158482

Academic Editors: Kris Law, Yuk-Ming Tang and Jukka Majava

Received: 14 June 2021
Accepted: 21 July 2021
Published: 29 July 2021

1. Introduction

The Internet has revolutionized the way in which information systems are conceived, and the Web has become the framework to publish and consume all types of multimedia contents and resources [1]. The impact of this new digital world has transformed the software industry [2] and also been reflected in educational environments and more now with the tendency of remote activities.

The Web provides a wide range of resources and services, one of them focused on the teaching/learning process, but it also causes several challenges. Educators must design new strategies not just for the classroom but also to cultivate out-of-classroom studying time with meaningful homework. There are some classical examples implemented as classroom activities or projects to show to the students the working of their field. However, there is a problem when these activities become classical: many solutions might be found on the Web. The existence and availability of these public solutions are a real challenge for teachers who want to reuse these specific learning materials, since valuable extra class time devoted to studying activities could end up as a mere copying exercise of someone else's work. Rehashing these activities will leave the teacher with no certainty that the students have learned their subjects. Hence, the current communication technologies bring also new risks and forms of non-ethical behavior that sometimes are not easy to detect [3].

The Internet and its impact on the education sector have encouraged the constant transformation of the teaching/learning strategies. The Web is reinvented every day, what was considered technically impossible to do, nowadays is not just possible, but easy, since the technologies used for their development have continued improving and innovations appear every day. Another current impact on the educational sector is the Coronavirus Disease (COVID-19) pandemic, which has affected more than 1.5 billion students worldwide and has exacerbated inequalities in education; the head of the ONU has affirmed that the decisions taken now in this regard will have a lasting effect on hundreds of millions of people and their countries' development. (UNESCO study about COVID-19, available via https://en.unesco.org/covid19/educationresponse/globalcoalition (accessed on 20 July 2021)). Some results have shown that traditional schemes must be extended at distance modalities, not doing that would result negatively on the quality and could increase inequality of learning opportunities [4].

This era, and more with the Coronavirus Disease (COVID-19) pandemic, demands and challenges teachers to reform their traditional courses and more in technological areas as Educational Software Engineering, even more, if the subject is related to programming or developing web applications. Web development, a fundamental topic in Software Engineering and Cloud-based applications, deals with various critical challenges in education such as learning continuous emerging technological tools, plagiarism detection, generating innovative learning environments, among others. So, teaching and learning web applications is a changing topic that cannot remain static.

Research has been made to emphasize the difficulty of teaching and learning programming [5–11], and also expressing the main challenges of teaching web programming [12–15]. Some of these works have proposed different strategies to deal with the challenges; for example, in the effort required to acquire a full basic training, some authors determine that a single college course is enough [13,16] but with binding prior knowledge. Others think that two courses are the minimum [12]: front and back-end respectively.

Derived from the above, the following questions were raised: (a) is it possible to prepare a web programming engineering-level course building the skill set of the students guiding the learning process with the development of a common specific solution but avoiding plagiarism?; (b) is it possible for this course to append a customized experience for every student's needs and requirements? and; (c) given the fast pace of current technological changes, is it possible to obtain a course that does not become obsolete every passing course?

The aim of this paper is to introduce a sustainable teaching/learning methodology for web programming courses in Engineering Education using project-based learning adaptive to the continuous web technological advances. The above three questions are answered throughout the paper and clearly emphasized in the Discussion section. It also provides facilitators, and even students, a different strategy to deal with subjects with high technological content as Web Programming. The methodology combines the knowledge and experience of the teacher to guide the students for the development of an integrating project. This project is generated through student iterations (sprints) during a complete course, making it a personalized experience from the beginning to the end of the course, avoiding plagiarism.

This is a descriptive research paper which explains the methodology, shows an evolution of the web technologies that have been taught in past and more recent courses, and provides surveys to know the opinion of the students. The methodology has been continuously improved over the last 9 years, with the benefits of being adaptable, proved, and personalized to cover the needs of every student. Even more, it was successfully applied in our last course despite that it was taught in a distance modality. Although the methodology has been applied in different scenarios, here we also present a syllabus example of how to adjust a course with 48 contact hours and 48 non-contact hours. In the example, we explain how the students acquire their proficiency.

The paper is organized as follows: First, we give a brief summary of the difficulty of teaching/learning programming, the challenges of teaching web programming, and some strategies implemented by other researchers. Then, we explain how we have taught an introductory Web Programming engineering-level course attending the modern requirements and using the PBL approach. Later, we describe the experimental process carried out to improve the methodology. In this section, we explain the process of various years of polishing the methodology, the students and universities involved, and argue in what sense it is adaptive to new technologies. Then, the result of years improving the integrating project is described with a syllabus of web programming and its methodology. Afterward, we exemplify our methodology with our last course (carried out remotely because of the coronavirus pandemic problem) emphasizing the different activities together with the concepts that students are expected to learn to acquire their proficiency. In addition, we discuss the benefits of using this methodology and give our conclusions in the last section.

2. Difficulties of Teaching/Learning Programming and Challenges of Web Development

Many works have been written about the difficulties of teaching computer related subjects. In particular, this section will discuss some works exposing the difficulty of teaching and learning programming in general. Then, we focus on explaining the challenge of teaching web programming. Lastly, we discuss those works proposing different strategies in teaching web programming and web development.

2.1. The Difficulty of Teaching and Learning Programming

Teaching and learning programming is a difficult task that requires a lot of effort, dedication, training, and mathematical knowledge, among other skills. So, why is computer programming so difficult to learn? Bosse et al. [17] consider that programming is a crucial part of software engineers training and trying to identify difficulty patterns related to learning how to program. Piteira and Costa [6] study the difficulties in learning programming using teachers and students' opinions and exam results. They identified that the difficulties arise in a lot of aspects: the contents of the programming topics, limited practical sessions, material presented by teachers, etc.

The complex combination that arises from the described situation is a serious problem that causes lack of motivation and eventually, the abandonment of a course. Figueiredo et al. [10] comment that it is important to act as soon as this discouragement is noticed; the follow-up of each student must be immediate and personalized. They affirm that it is possible to build a profile of each student's competences and skills in introductory programming to record their improvement and encourage growth.

Some authors have focused on proposing new ways of teaching programming skills [7]. Others try to understand how to awaken computational thinking [8,9,11,18,19]. The programming difficulty encompass other paradigms, like Object Oriented Programming [5] and Web Programming, as emphasized in the next subsections.

2.2. The Challenge of Teaching Web Programming

Wang and Zahadat [12] conducted a study where they emphasize that the explosive growth of Web 2.0 Technologies present a significant challenge for teachers focused on teaching web development. In other studies, they presented an IT educator's perspective and describe some challenges and problems of teaching Web development. Some of the approaches that they presented are: Web development to user interaction, client-side and server-side Web development, real-world applications, and constructivist teaching methods.

In a very similar sense, Xinogalos and Theodore [14] make an important study about the main challenges a professor can find when teaching web programming. Liu and Phelps [13] present a similar study in terms of challenges and also include important tools that should be used in the teaching process of web programming. They focus on a junior level course, similar to our case, but they are more busy teaching the foundations of web programming and sharing their experiences than the technologies themselves.

All previous authors agree that teaching web programming is a challenging task. Douce [15], for example, carried out a study about understanding the tutor's perspective in the teaching of web technologies, interviewing 12 teachers and agreeing that the issues of programming, solving problem skills, and including several technologies in a course, is a challenging topic not only for students but also for teachers.

Guo and Koufakou [20] summarize the educational problem with respect to the complexity of web involving cloud computing technologies. It involves vast technologies that teachers need to cover such as high-performance computing, distributed systems, networks, databases, security, data analytics, etc. and includes web mobile development in the curriculum [21].

Connolly [22] provides a historic overview of the key knowledge areas potentially needed by full-stack (involving front-end, back-end, and DevOps) from 2008 to 2018. He shows that current full-stack developers need to know recent concepts and technologies such as frameworks in the front-end and back-end; cloud native architectures; web sockets; APIs; reactive programming; parallel coding patterns, etc. However, he comments that one of the key teaching problems in any computing program is that concepts, techniques, and technologies are taught within separated courses (e.g., databases, networks, programming), but in industry, they are parts of larger heterogeneous environments. Students must be competent and be able to put all knowledge together into a single application [23].

2.3. Teaching Strategies in Web Programming

Wang and Zahadat [12] have developed a method consisting of four elements: (a) Concentrate on AJAX; (b) Divide the course in two: one focused on the client and one on the server; (c) Assign projects that integrate topics with real world applications; and (d) Use constructive teaching methods. Although what they propose is very interesting and useful, some curricula include a single subject related with web programming. Susmita Kar et al. [24] also have reformed their curricula in a two-semester web course; they consider it to have a huge impact on achieving the minimum required skill in the web development field in the software industry.

Rosenbloom et al. [16] propose a twelve-week course that involves starting with Model View Controller (MVC) frameworks and including web service topics as RESTful but specifically specifies that students should have prior basic knowledge.

Connolly [25] has proposed that dedicating several courses to web content is a good start; another approach consists in integrating web practices throughout the curriculum. Maiorana [26] discusses a case study based on the administration of a logging system, covering the main topics of web programming, databases (transactions, stored procedures), and security aspects. We take into account, partially, the Maiorana methodology to design the basic framework of our practices. However, we do not take into account any security aspects in depth as this is not the focus of the course.

We know that it is not easy for students with little experience in web programming to absorb so much information in a short period of time. However, one of our objectives, in a first course, is teaching the students fundamental concepts of front and back-end web application development through the learning process of putting all knowledge together into a single application using current tools so that in the future when new web technologies appear they can quickly migrate to the new ones.

3. A Web Programming Engineering-Level Course

Web Programming is a topic with high technological content and maybe one of their main purposes is either for programming jobs or for future courses that require web programming. Next subsections start by explaining the role of the web developer in the industry and what should be the main objective of a first web programming course; then we explain a case about how we have taught it by employing project-based learning.

3.1. Fundamentals

Although web programming is, in some cases, only one or two courses within a computer science curricula, the role of the web developer in the industry is itself a professional trade. To build websites and web-based apps, web developers work with Markup languages, high-level programming languages, libraries, frameworks (front-end or back-end), and database. Web developer's jobs may be split depending on whether they are working as a front-end, back-end, or full-stack developer. Full-stack developers specialize in both the front-end and back-end. Jordan Shropshire, et al. [27], give a more complete definition about what is a full-stack developer. Considering the above, understanding front-end and back-end concepts are the backbone part of Web Development.

Web programming and web development are not the same, as mentioned by Connolly [22]; one of the key difficulties while teaching web development is that it has a large amount of potential topics, more than can possibly be covered in a single course. A decade ago learning web programming meant to know HTML, CSS, and a back-end language programming. However, now, web development means to know web programming and also learning frameworks, MVC, AJAX, different protocols like HTTP, TCP, FTP, etc; server configuration, web services (WSDL, REST), security aspects; database access, docker, etc. The reality is that being a web developer requires knowing several courses and one of them is Web Programming.

Wang and Zahadat [12], proposed the division of a web programming curriculum in at least two courses: one focused on the client side and the other one on the server side. We also propose two programming courses, but with the following strategy: the first one covering generalities, concepts, the technologies currently in use, knowing front-end and back-end, and being proficient in web programming. The second one covers specific topics like front-end frameworks to build user interfaces, Model View Controller (MVC) Frameworks, and Web Services. Depending on the curricula, it may require other helper courses like computer security, databases, service oriented architecture, etc.; but it depends on the undergraduate career profile. This advanced course or helper courses will not be the subject of this paper and we will focus on the beginner one.

3.2. Project-Based Learning

From a teacher's perspective, it is easy to monitor students' growth by employing continuous evaluation. However, how can we engage students to be interested in increasing their own knowledge and skills? Project-Based Learning (PBL) is a good strategy and more in selected topics of computer science. Fioravanti et al. [28] have reported that applying PBL in Software Engineering topic has led to students being more enthusiastic and positive. In addition, as discussed by García-Peñalvo [29], in the education innovation indicators give PBL as one of the suggested strategies to deal with the new generations.

Sometimes PBL is confused with problem-based learning. The second one sometimes is implemented by teachers in partials or at the end of the semester. The first one is more directed to the application of knowledge, whereas problem-based learning is more directed to the acquisition of knowledge [30]. PBL is more complicated to apply than problem-based learning because it might be implemented during a complete course or between different subjects. One of the difficulties lies in how to merge the project with the syllabus of the subject(s).

In subjects with high technological content, developing competences in students is more beneficial when concentrating all knowledge together into a single application [23]. So, we have used PBL in our suggested course.

3.3. Remote Desktop Application as an Integrating Project

Ten years ago, a course for developing a web interface project very similar with an existence operating system was proposed [31]. Such a project started not covering a complete course but only including some laboratory practices. Later, it evolved to be an integrating project, which has been applied from the beginning until the end of a course. The integrating project consists on the development of a Remote Desktop Application that emulates a commonly used Operating System where instead of an interface-to-hardware communication as is the case in a normal operating system, the interface has communication and functionality through HTTP based communication.

We have found that the familiarity of the students with the operating systems' design given by everyday use of computers was an advantage, as the students could focus their efforts on the Web Programming concepts and it did not add a layer of cumbersome extra challenges to the task, so the students could focus on learning web development.

An Operating System (OS) is special software that allows a user to take advantage of the hardware of a computer. An OS has two principal purposes [32]: (i) Present an environment where the user can run different programs; and (ii) Control the various hardware parts that form the computer and allow the user programs to run. The two OS purposes are very easy to differentiate and identify as separate entities, i.e., a normal computer user does not need to know the details of how the OS calculator made an arithmetic operation and how the result appears on the screen. Because a student in computer science topics must be familiar with at least one OS, he could intuitively understand how the OS should interact with the user without even knowing how the OS is interacting with the hardware.

Even more, the part of the operating system that interacts with the user has some very convenient elements to make the interaction as frictionless as possible between the operator and the machine. Some examples of these interactive elements include icons, bars, windows, shortcuts, and commands. These elements vary between OS vendors and between different versions of an OS from the same vendor.

Given the clear separation achieved between the front and back-end part, it makes sense to propose a Remote desktop based on an existing OS as an integrating project for the web course. The student with this project will have the freedom to choose the specific look and feel of the interactive elements and simulate different common programs normally available to an OS user, delivering a good number of learning activities that can easily cover the full content of a normal course.

The integrating project has required a student to choose an existing OS in order to replicate it in terms of web design (front-end) and to develop some modules so that a simulation is actually reached in terms of operability and good design.

Figure 1 shows a Data Flow Diagram (DFD) about the general final-user perspective of the Remote Desktop based on an existing OS. The dotted rectangle states a general login process, following the idea of Maiorana with respect to the login system [26]. Once a user authenticates, the desktop is shown. Then, an application option can be chosen by the user as many times as he wants, until a shutdown application has been chosen. Although our focus is a Remote Desktop Application, the DFD can illustrate any website system. The teacher participation has consisted in teaching concepts, guiding the students in the process of designing and developing the project.

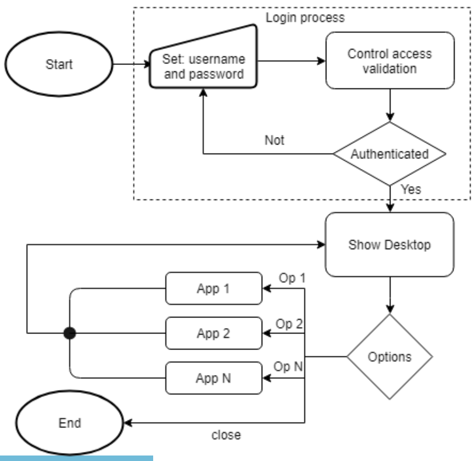

Figure 1. Data flow diagram about a general front-end perspective of the Remote Desktop.

4. Experimental Process

Our experimental process is described from three main standpoints: First, the students involved in receiving the teaching and learning strategy of the Remote Desktop as an integrating project described in Section 3.3, from when it has been applied, statistics on the compliance, and some recommendations. Second, statistics about all front and back-end technologies taught from the first course when PBL was applied until the time the paper was written. Finally, statistics about the opinion of the students who have received this teaching/learning strategy.

4.1. Experimental Universe and Applied Evidence

Our teaching/learning methodology has been applied for 9 years with 15 different groups and in three different colleges: 5 at Universidad Panamericana (UP); 4 at the Technological Institute of Tuxtla Gutiérrez (ITTG); and 6 at the Polytechnic University of Chiapas (UPCh). See Table 1 for a statistical summary; the approved column signals when the students were able to demonstrate their knowledge and deliver their project, having their evidence accepted. Note that, in recent courses (at the top), the approval percentages (last column "%") were better compared to the last one. Gradual changes were implemented with each iteration of the course; these include adjustments in topics' content due to the emergence of new technologies; differences in the duration of each applied course (see column Grade in Table 1, **q** quarter or **s** semester); and student's evidence, for example, columns Initial Test (IT), Middle Test (MT), Final Test (FT), Laboratory Practices (LP), and Integrating Project (IP) denote the activities that students had to deliver in that course. You can see that, in the last 6 courses the methodology has been stabilized, including the last course that was imparted in remote mode due to the COVID-19 pandemic problem.

Table 1. Students receiving the methodology organized by universities.

Group Name	Grade	Evidence					Num. Students	Approved	Not Approved	University	%
		IT	MT	FT	LP	IP					
PW20	5s		x	x	x	x	20	20	0	UP	100%
PW2	5s		x	x	x	x	16	16	0	UP	100%
PW1	3s		x	x	x	x	21	18	3	UP	86%
DAW6344	3s		x	x	x	x	14	14	0	UP	100%
DAW6345	5s		x	x	x	x	16	14	2	UP	88%
A218	9s		x	x	x	x	30	28	2	ITTG	93%
A217	9s			x	x	x	18	14	4	ITTG	78%
B217	9s			x	x	x	27	26	1	ITTG	96%
C217	9s			x	x	x	30	22	8	ITTG	73%
A216	6q		x	x		x	30	21	9	UPCh	70%
A215	6q		x	x		x	40	28	12	UPCh	70%
A213	6q	x	x	x		x	36	22	14	UPCh	61%
A212	6q	x	x	x		x	24	14	10	UPCh	58%
A211	6q		x	x	x		34	23	11	UPCh	68%
A209	6q		x	x	x		23	14	9	UPCh	61%

On the other hand, we can observe that groups imparted in UP and ITTG have the best approval index. Although we must clarify that the courses given in these Universities have longer duration than in the UPCh. In addition, in the semester when this course was imparted over ITTG, the students had already a previous database course, whereas the ones from UPCh or UP (particularly DAW6344 and PW1) had not. Another clarification is that over UP, the students did not have any previous basic network training, so the students were a bit hampered when confronted with the client–server model.

Our analysis of the data showed that, in the case of UPCh, it was important to divide the course in two, as suggested by [12,33], and more because the duration course was by quarter. In the case of ITTG that already had two courses focused on the development of web applications, it was recommended to students to conclude their Web Programming course before taking their Advanced Topics in Web Programming Technologies course. In the case of UP, we have suggested including a Network and Database course before the Web one and to split the course in two as was mentioned above.

Making these changes will open the possibility of including in advanced courses another methodologies course. For example, Harriger and Woods [34] proposed a teaching method based on the development of websites for local businesses, together with a web-based software development methodology. Another work that suggests something similar is that of Margaret et al. [35], where the development of a portal web with teams of two or three students is proposed.

4.2. Technologies and Desktop Operating Systems Worked in All Courses

Table 2 illustrates historical web technologies that worked in each of the 15 groups. In the upper part, you can see the year which each of the groups corresponds to. The table shows the front-end and back-end technologies (first two columns). We can observe several technologies that were previously seen in the courses but not anymore, such as ASP and JSP. In addition, in the table it can be seen that MySQL is the best Database Server technology we have taught, which made us think that we could integrate other object-oriented technology such as MongoDB, which you can see in the last groups.

PHP and JavaScript Server (Node.js) have been the most recent server programming languages we have used in these courses. Note that these languages have not been taught at the same course, since in the back-end part of the project, students were left free to choose the programming language; this has caused some students to select Node.js and .NET. In this sense, we have observed that only 10% of the students prefer to use some technology different from the one proposed in class.

Another analysis is reported in Table 3, which illustrates a relation of all desktop operating systems that worked in each of these groups. The first column of the table shows a category of the different operating systems (based on Windows, Linux, MAC, Mobiles, and others); the second column describes the desktop operating system, in some of them it is defined a specific version; the rest of the columns specify how many students have chosen such an operating system. From the table, we can see that when the group is very large students have repeated the OS and the risk of plagiarism (at least in the front-end topics) is bigger. On the other hand, when groups are smaller the OS chosen are always different; therefore, the risk of plagiarism is lower. It is normal and even desirable for classmates to share and help each other over the course. This is not exclusive to current classmates; this is the case even from past students of the course. It is inevitable that sometimes students will share source code for their solutions. Unlike other areas, at least from our experience observation, to reuse source code is in fact a good engineering practice; it is important that students know how to adapt good code of another solution to their own solution.

The selection process of the desktop operating system has always been free for the students. However, sometimes the students want to choose the same one; the suggestion has always been that they look for another alternative or at least not to repeat the same desktop operating system by 3 students or more.

Table 2. Historic front-end and back-end technologies used on each generation.

		2011	**2013**	**2014**	**2015**	**2016**			**2017**			**2018**			**2019**	**2020**
	Technologies	**A209**	**A211**	**A212**	**A213**	**A215**	**A216**	**C217**	**B217**	**A217**	**A218**	**DAW6345**	**DAW6344**	**PW1**	**PW2**	**PW20**
Front-End	HTML	√	√	√	√											
	HTML5					√	√	√	√	√	√	√	√	√	√	√
	CSS	√	√													
	CSS3			√	√	√	√	√	√	√	√	√	√	√	√	√
	jQuery				√	√	√	√	√	√	√	√	√	√	√	√
	Angular.js						√	√	√	√	√					
	Boostrap						√	√	√	√	√	√	√	√	√	√
	Material design							√	√	√	√					
Back-End	Tomcat	√		√	√											
	Apache					√	√	√	√	√	√	√	√	√	√	√
	IIS		√						√	√						
	Node.js							√	√	√	√	√	√	√	√	√
	MongoDB										√	√	√	√	√	√
	MySQL	√		√	√	√	√	√	√	√	√	√	√	√	√	√
	SQL Server		√													
	ASP		√													
	JSP	√		√	√											
	PHP					√	√	√	√	√	√	√	√	√	√	√
	Server JavaScript							√	√	√	√	√	√	√	√	√
	NET								√	√						

Table 3. Desktop operating system assigned to the students.

		2011	**2013**	**2014**	**2015**	**2016**			**2017**			**2018**			**2019**	**2020**
Category	**Desktop OS**	**A209**	**A211**	**A212**	**A213**	**A215**	**A216**	**C217**	**B217**	**A217**	**A218**	**DAW6345**	**DAW6344**	**PW1**	**PW2**	**PW20**
Windows	Windows 3.11	1	2	1	2	2	1	1	1		1			1		
	Windows 95	2	2	1	2	2	2	2	2	1	2	1	1	1		1
	Windows 98	2	2	2	2	2	2	2	1	1	1			1	1	1
	Windows XP	2	2	2	2	2	2	2	1	1	1	1	1	1	1	1
	Windows 7	1	2	1	2	2	2	2	1		1	1	1	1	1	1
	Windows 8	2	2	1	2	2	2	2	1		1					
	Windows 10						1	1	1	1	1					
	Windows NT	2	2	1	2	2	2	2	1	1	1			1		1
	Windows me								1	1	1					
	Windows Vista								1	1	1					

Table 3. *Cont.*

Category	**Desktop OS**	**2011**	**2013**	**2014**	**2015**	**2016**	**2017**					**2018**		**2019**		**2020**
		A209	**A211**	**A212**	**A213**	**A215**	**A216**	**C217**	**B217**	**A217**	**A218**	**DAW6345**	**DAW6344**	**PW1**	**PW2**	**PW20**
Linux	Debian	2	2	1	2	2	1	1	1	1	1	1	1	1	1	1
	Ubuntu	2	2	2	2	2	1	1	1	1	1	1	1	1	1	1
	Linux Mint	1	2	1	2	2	1	1	1		1		1	1		
	Red Hat	2	2	2	2	2	1	1	1	1	1	1		1	1	1
	Fedora								1		1	1	1	1	1	1
	CentOS		1	1	1	2	1	1	1		1	1		1		
	SUSE		1		1	2	1	1	1	1	1					
	Linux Arch								1	1	1					
	Manjaro									1	1					
	Kali											1		1	1	1
MAC	OS	2	2	1	2	2	1	2	1		1					
	OS 9	1	2	1	2	2	1	1	1	1	1	1	1	1	1	
	OS X		1	1	2	2	2	1	1	1	1	1	1	1	1	1
	Solaris													1	1	1
Mobile	Android		1	1	2	2	2	2	1	1	1	1	1	1	1	1
	OS movil		1	1	1	2	1	1	1	1	1	1	1	1	1	1
	OS for Ipad											1	1	1	1	1
	Blackberry	1	1	1	1	2	1	1	1		1					1
	Window Phone										1					
Others	Sun OS 4.1		1	1	1	1	1	1	1		1					
	Sun OS 5.1		1	1	1	1	1	1	1	1	1	1	1		1	1
	Firefox OS										1	1	1	1	1	1
	Nintendo Switch OS													1	1	1
	Xbox OS														1	1
Students/group		23	34	24	36	40	30	30	27	18	30	16	14	21	18	20

4.3. Students' Statistical Opinion

Aiming to analyze the impact the development of the Remote Desktop Application had created in the students, in 2019 we applied a survey to those who had received a course with this methodology. It does not includes the last course (2020); exact questions and answers can be visualized via: https://git.io/JegmF (accessed on 20 July 2021). Table 4 shows the general aspects of the survey. From the 369 students that had taken the course until 2019, we had a representative sample of them, 68 opinions, which will be analyzed below. (Note that those 20 students of PW20-2020 course were not included in this survey because they will be part of another one, explained later).

Table 4. General aspects of the survey.

Description	Sub-Description	Sub-Total	Total
Number of questions			10
	Open-ended questions	2	
	Closed-ended questions	8	
Students receiving the methodology			379
Students sent the survey			245
	Survey sent via email	214	
	Survey sent by social network	31	
Students answering the survey			76

Table 5 illustrates a general opinion of the students with close-ended questions. Questions 1–5 had 3 possible answers: [yes, no, a little]. Question 6 only had two possible answers [yes, no]. Question 1 and 2 were included to reflect on how they felt in the past, Questions 3–5 were raised to reflect on the present considering an event of the past. With Question 6 we wanted to identify the lingering opinion left after their graduation. From the results shown in the table we conclude that although the students felt overwhelmed, they were left with a lot of positive opinions, including: the project was good intellectual challenge, it helps to strengthen their Web knowledge, and that it was a good self-learning strategy. Derived from Question 1, we concluded that we should improve and adjust the workload.

Table 5. Closed-ended questions (possible answers: yes, not, or little).

Number	Closed-Ended Questions	Answer					
		Number			Percentage		
		Yes	Not	Little	Yes	Not	Little
1	Did you feel overwhelmed in the development of this application during the course?	10	20	38	15.50%	29.30%	55.20%
2	Developing the Remote Desktop generated you an intellectual challenge?	64	2	2	93%	3.50%	3.50%
3	Do you think that the exercises you were doing helped you to strengthen your knowledge on web programming issues?	63	3	2	91.40%	5.10%	3.50%
4	Do you consider that the development of the Project helped you as a self-learning strategy, taking into account that the rest of your classmates had a different front-end simulator to perform than yours?	65	2	1	94.80%	3.50%	1.70%
5	Do you consider the development of the Remote Desktop in a first web programming course a good strategy?	56	5	7	82.80%	6.90%	10.30%
6	Are you graduated?	44	24	N/A	64.70%	35.30%	N/A

Figure 2 illustrates three graphics: (a) A perspective on the total number of students who have worked with the integrating project and divided by universities; (b) A perspective on the total number of students who answered the survey and which universities they belong to; and (c) According to the answers, in which year they received the course.

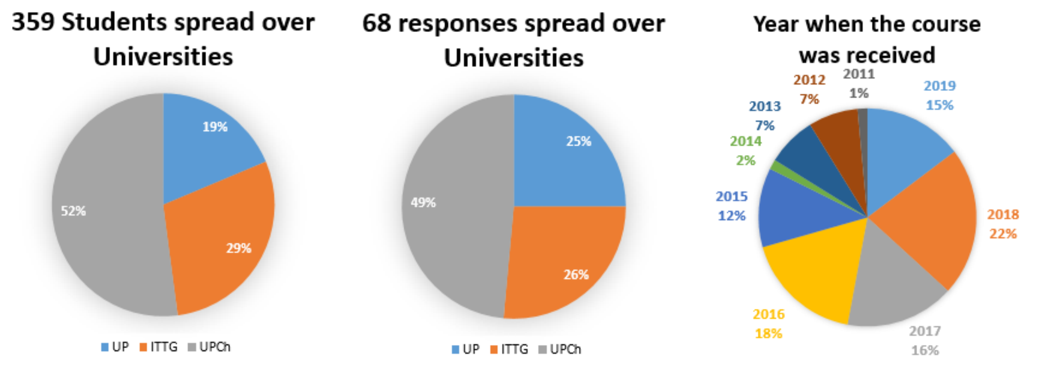

Figure 2. Left: total number of students; center: responses to the survey; right: responses spread over years.

Finally, two open-ended questions: (a) *What was your first impression when you found out you were going to develop a Remote Desktop Application as a final project?*; and (b) *Final comments*. With these two questions, we tried to identify some unknown aspects that students could notice. For Question (a) we extracted some keywords, highlighting: *Emotion*, *Challenge*, *Interesting*, *Afraid*, *New knowledge*. For Question (b) we found remarkable and positive answers (especially from graduated students) but also some negative answers (especially among still not graduated students). Some suggestions were received like using server-side frameworks, although they also considered that those topics could be overwhelming for an introductory course.

4.4. Students' Opinion of the Last Course

Table 6 reports the results obtained with a survey applied with our last course (PW20, Table 1). This last survey is based on those applied in Table 5, omitting the last question because it does not apply; the survey was answered by 18 students. From the results shown in the table we conclude that although the project is already not so overwhelming, it is a variable that must be considered when a course is relatively short. Derived from Question 1, we concluded that effectively, with every new course, we have improved the methodology and adjusted the workload in a good direction. From Questions 2–4, we can conclude that most students think the project was a good intellectual challenge, it helped to strengthen their Web knowledge, and that it was a good self-learning strategy. From Question 5, we have identified that although most students think that the integrating project is a good strategy, a few students do not like the strategy yet.

Table 6. Students' opinion of the last course.

Question	Answer					
	Number			Percentage		
	Yes	Not	Little	Yes	Not	Little
1	3	7	8	16.67%	38.89%	44.44%
2	13	2	3	72.22%	11.11%	16.67%
3	12	2	4	66.67%	11.11%	22.22%
4	15	1	2	83.33%	5.56%	11.11%
5	13	2	3	72.22%	11.11%	16.67%

5. Result: An Integrating Project Included in a Web Programming Course

The integrating project which we have applied for nine years has been improved course after course. We consider two main results: (a) A syllabus for an Introductory Web Programming course; and (b) A method about how to teach and learn Web Programming by developing a Remote Desktop Application using PBL.

5.1. Syllabus: Learning Units and Learning Outcomes

The following syllabus that we propose is based on the following reasons: (a) Studies of teachers' perspectives [15,36]; (b) Modern computer science web curriculum [25,37]; (c) Industry requirements, emphasized in Section 3.1; and (d) Our experience of years of teaching web programming, reported in Section 4. Our restriction is to provide in a single course, web programming topics of front-end and back-end, similar to Dugan [38]. Hence we have conformed the following general units:

1. Front-end: Design and development of web pages;
2. Back-end: Internet, WWW, and HTTP;
3. Front and back-end interaction.

Table 7 abstracts the topics and technologies for each unit. The first unit achieves three learning outcomes: (i) Design of web pages on the client side using HTML5 and CSS3 technologies; (ii) Development of programs involving the use of a programming language on the client side (JavaScript) and access their functionality through HTML5 and CSS3; and (iii) Knowledge of different client frameworks used in the design phase (e.g., Bootstrap, Material design, etc.) and the libraries used in the client programming phase (e.g., jQuery). As mentioned before, to consolidate knowledge and skills in front-end base technology, modern technologies to build user interfaces based on JavaScript, like React and AngularJS, will be left for a second more advanced course.

Table 7. General units, topics, and web technology examples.

Unit Name	Topics	Example's Technologies
1.-Front-end: Design and development of web pages	Web pages design	HTML{1–5}, CSS{1–3}
	Front-end programming language	JavaScript, HTML5 and CSS3
	Client Technologies for design and development	jQuery, Boostrap Material design
2.-Back-end: Internet, WWW and HTTP	Internet, TCP/IP and WWW	Browsers, URL and HTML Source code
	Client/Server architecture, and HTTP protocol	Browsers Developer tools
	Web and database Servers	Apache, Node.js, IIS, Tomcat, MySQL and {L,W,X,M}AMP
3.- Front and Back-end interaction	Server-side web programming languages	ASP, JSP, PHP, JScript Server
	Web programming paradigm in the server side	Session variables, cookies, AJAX
	Develop programs using a server-side language	Selected Server Programming

The second unit achieves the following learning outcomes: (i) Description of the historical perspective of the Internet, the family of TCP/IP protocols, and World Wide Web (WWW); (ii) Knowledge of the client/server architecture focusing on the HTTP protocol mechanisms; (iii) Web servers and databases.

Finally, the third unit achieves three learning outcomes: (i) Identification of existing technologies for server-side web programming; (ii) Knowledge of the paradigm of web programming on the server side; (iii) Development of web programs using a server-side language.

We propose teaching one of the active server programming technologies: PHP, ASP.NET, JavaScript server, as shown in Table 7, or any other currently active technology (the selection of the server language programming is transparent to the methodology and the decision is left to the instructor according to their experience). Regardless of the language being used, the web programming paradigm (client and server execution; cookies and database connection; session variables and AJAX) should not vary. With these three units we cover the main topics of an introduction to web programming. To not saturate the students and considering that it is a first course of web programming, we have decided to include additional topics like Model View Controller (MVC) Frameworks in other more advanced courses.

5.2. A Method for Teaching and Learning Web Programming Using an Integrating Project

When starting a new course, a teacher, once knowing the syllabus, asks the question about how the course will be evaluated and taught. The next subsections explain, first, the evidence classified by a type of proficiency and then how such evidence must be developed and delivered using PBL.

5.2.1. Proficiency

Programming skills, computer programming requires many cognitive skills. Thinking and problem-solving skills are some of the core skills required by students for learning programming [39]. However, one of the benefits of computer programming is also nourishing the problem-solving skill [40].

Knowledge, within a Competency-Based Assessment, knowledge is defined as a competence that students must acquire during a course. According to the first two levels of Bloom's cognitive domain, knowledge and understanding are basic and very important skills that a student must possess, since the learned concepts serve as a platform to come into new knowledge [41]. Walraven et al. [42] carried out a study about how students acquire knowledge and solve information problems and what kind of criteria they use when evaluating results.They conclude that students spent most of their time searching and scanning and only a small fragment of time processing and organizing information. This compromises their level of knowledge.

To achieve a good balance between theoretical and practical knowledge as grounded above and taking into account our experience of years of teaching web programming (reported in Section 4, in particular, deductions of Table 1) and the students opinion, the activities and projects we propose are classified into two types of competencies: (a) Knowledge (middle and final test), and (b) Skills (PBL and laboratory practices). Table 8 abstracts eight pieces of evidence classified in these competencies.

Table 8. Evidence of knowledge and skills for our proposed Web Programming subject.

Proficiency	Evidence Classification	Evidence
Knowledge	Theoretical Test	Middle Test
		Final Test
Skills (Integrating Project)	Laboratory Practices	Front-End
		Back-End
		Front and Back-End
	Project	Front-End Design
		Setting Back-End
		Front and Back-End Development

5.2.2. The Process of Integrating the Remote Desktop Application with PBL

The integrating project: after years of polishing the integrating project explained in Section 3.3, we propose the strategy of guiding the students in developing the Remote Desktop Application following the proposal illustrated in Figure 3. The methodology is divided vertically (time representation) into four blocks: Block 1 involves the initial part of the project with front-end topics such as HTML, CSS, and JavaScript (Unit 1 in Table 7). Block 2 includes Back-End technologies (Unit 2 in Table 7). Block 3 and 4 includes Front and Back-End interactions (Unit 3 in Table 7); these blocks are the final part of the project and can be completed once the back-end technologies have already been taught.

Figure 3 also shows, horizontally, the evidence described in Table 8. As can be seen in the red rectangle, the development of the project covers the entire course. It starts by requiring that each student chooses an existing Operating System (Block 1 in Figure 3, as it is also explained in Section 3.3). The choice must be replicated by the student in terms of its front-end design. The main aim, in this stage, for the students is to learn front-end technologies by having a reference in good design (an operating system already consolidated). In general, software design is a complex cognitive process [43]. Reaching a good user interface is also complex and important, because it can facilitate interactions between the user and the application. User experience is quite an intricate field that involves anticipating the user preferences and then creating an interface that understands and fulfills those preferences. The user experience not only focuses on aesthetics but also maximizes responsiveness, efficiency, and accessibility of a website. Considering that designing a good user experience is time consuming, proposing a new appearance and interface is not the objective of this project. Instead, we prefer the students to challenge themselves to reach the design already established. The project demands students to stick as closely as possible to the design of the operating system they have chosen.

Then, client/server concepts are explained to the students and they must configure the different servers their project requires (Block 2 in Figure 3). The final project description starts by proposing which server programming language they prefer to work with (Block 3 in Figure 3). It is important that each student makes an opinionated decision in advance. With this decision, the teacher achieves a double goal, first, to make them see advantages and disadvantages of the different technologies, and second, to minimize the risk of copying between classmates.

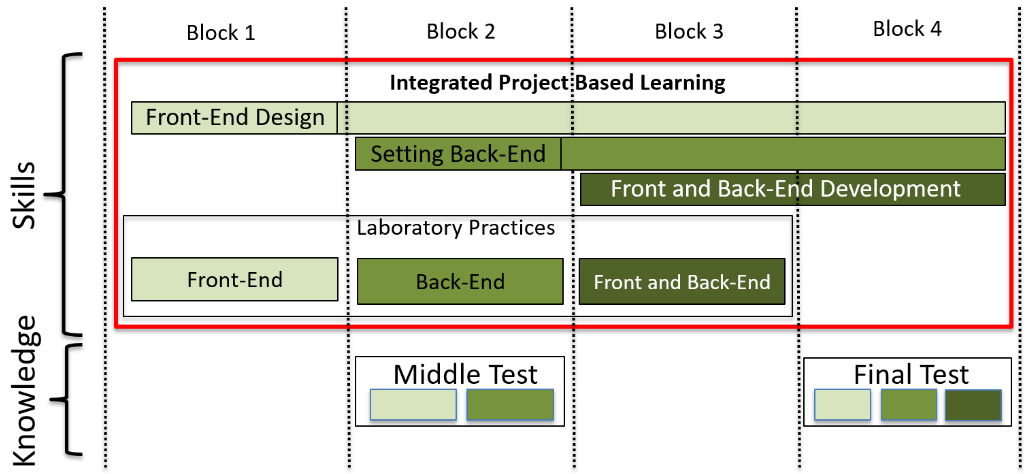

Figure 3. The methodology illustrating the PBL within a red rectangle embedded in a course divided vertically by blocks and horizontally by competencies.

Laboratory practices: Grade must measure personal skills and growth, so we divided the project into small grading units or laboratory practices and we kept a record of their development status. These practices stack over time and build the foundation of the integrating project. We have broken up all practices into three blocks, in this case, they are also the three learning units of the syllabus: (a) Front-end; (b) Back-end; and (c) Front and back-end interaction. In Figure 3, the laboratory practices (within skills evidence) are made in blocks 1 until 3; these make synergy with almost the whole integrating project. The last block is left without laboratory practice to reduce workload in the students and they can prepare their final delivery (a detailed example of laboratory practices is explained in Section 6.2).

Acquiring Knowledge: one of the instruments used to identify the acquisition knowledge competency is performing a test, which can be theoretical and/or practical. The theoretical tests usually measure the memorization of concepts, but it will not work to assert the acquisition of the skills required to solve pertinent problems. Usually practical tests give evidence about the achievement of some level of expertise. However, practical tests have time constraint problems: some students may be able to finish in the time slot assigned and some may not, but this may measure their agility more than the depth of their skills. Regardless of the type of test applied to the student, they are important as they serve to identify possible weaknesses that the students might still have. After the exam, we can work with them to remove those weaknesses and to know if they have reached their knowledge proficiency.

Hence, (and taking into account the experience portrayed in Table 1) two exams are proposed: midterm and final (Blocks 2 and 4 in Figure 3 respectively). The midterm exam includes concepts about front-end and back-end (internal rectangles with light green and solid green), and teacher must identify those students' weaknesses. The final exam has the objective of demonstrating significant knowledge of front-end, back-end, and understanding on the client/server architecture, HTTP protocol, configuration ports, session variables, cookies, and AJAX (light green, green, and dark green) and also identifying if students demonstrate proficiency in those weaknesses identified in the midterm exam. It can also be noticed in the figure that the exams are outside the scope of the integrating project but integrated into the total evidence. This means the project development supports the enforcement of theoretical and practical knowledge.

6. Matching the Integrating Project with an Introductory Programming Course

To explain how the syllabus is coupled with the integrating project and give certainty about the starting up of our proposal, we will use our last course, which was 100% remote because of the Coronavirus Disease (COVID-19) pandemic. This course, so-called *Introduc-*

tion to Web Programming, comprises 16 weeks, each week involving three face-to-face hours and three extra-class hours. Table 9 details each piece of evidence (describing laboratory practices) with an estimated time inside and outside the classroom.

Table 9. Detail of all pieces of evidence applied in the example course of web programming.

No.	Unit	Evidence	Description	Hours in Classroom	Hours Outside The Class
1	1		Initial Rules of the Course	1	1.5
2	1		Explanation: WBRDA Front-end	1	
3	1	FEP	Identity	1	1
4	1	FEP	Login	3	4
5	1	FEP	Calculator	3	4
6	1	FEP	Editor	2	4
7	1	IP	WBRDA Front-end	3	4
8	1		Challenge practice	1.5	
9	2	BEP	Install Web Server	1.5	1.5
10	2	BEP	Client vs. Server	2	2
11	2	BEP	Install Database Server	2.5	3
12	1 & 2	ME	Middle Exam (Front-end)	1.5	2
13	1 & 2		Challenge practice	1.5	
14	3		Explanation: WBRDA Back-end	1	
15	3	CSP	Authentication module	4	5
16	3	CSP	Block notes	4	4
17	3	CSP	Users Manager	4	4
18	3	CSP	File explorer	4	4
19	1, 2 & 3		Challenge practice	1.5	
20	3	FP	Project Integration	3	4
21	1, 2 & 3	FE	Final Exam (Front and Back-end)	2	0
			Sub total	48	48
			Total	96	

WBRDA = Web Based Remote Desktop Application; IP = Initial Project; FEP = Front-End Practices; FP = Final Project; BEP = Back-End Practices; ME = Middle Exam; CSP = Client–Server Interaction Practices; FE = Final Exam.

The next subsections explain in detail the different parts that compose the integrating project, how it is developed during the course, making synergy with the laboratory practices, and the knowledge tests we have applied.

6.1. The Integrating Project

It consists of three stages: (a) Initial project specification, the design part (Table 9, No. 2); (b) Back-end, configuring the server (Table 9, No. 9); and (c) Back-end implementation (Table 9, No. 14).

Designing the Remote Desktop: this stage requires every student to choose existing Operating System like a Linux Distribution or a specific version of Windows, macOS, Android, Windows Phone, iOS, etc. The choice is analyzed in terms of its front-end design. The complete list of activities of this stage, together with the knowledge that the students must acquire or apply, are shown in Table 10. Note that the table shows activities that can be matched one to one with the processes of Figure 1. With the development of the initial project, students will manage to acquire knowledge in a set of front-end technologies, (e.g., HTML, CSS, JavaScript, and derived libraries).

Table 10. Laboratory practices for the initial project and learning topics: H = HTML, C = CSS, JS = JavaScript, D&Q = DOM and JQuery.

No.	Practices	Description	H	C	JS	D&Q
1	Identity (App1)	Web page with student personal data and subject information.	√			
2	Login	Authentication mechanism with user and password options linked to a startup desktop as similar as possible to the selected OS, containing a menu which let navigate for different applications.	√	√		
3	Calculator (App2)	Calculator with a very similar appearance with the design of the selected Operating System. It must calculate basic arithmetic operations (+, −, *, /, %).	√	√	√	
4	Editor (App3)	A system able to add and delete notes.	√	√	√	√

Back-end, configuring the Server: Even though configuring web working tools for developers using technologies such as LAMP, XAMP, WAMP, and MAMP is relatively easy and also students demonstrate certain satisfaction when using any of these encapsulated technologies [44], installing one of these technologies can be a tedious task with no significant learning. The menial problems that usually arise from the installations can make the student ignore the more important job of learning the server concepts and techniques without the use of tools. These basic concepts include Web Server definitions, Database Management System (DBMS), FTP, etc.; it is an important part of the learning process to focus on much more important aspects such as configuration of ip, port number, session id, maximum amount of file uploads, etc., which are indeed part of the Web Paradigm. For this reason, this part of the project helps the students understand such concepts and provide continuity to the project integration but now scaling to the Remote Desktop.

Back-end implementation, the result of the final project will be a Remote Desktop HTTP-based protocol. Its development starts after the client/server concepts have been explained and students have carried out practices concerning installation of different servers (Table 9, No. 14). Students may propose which server programming language they prefer to work with. This triggers the students to explore different technologies and also minimizes the risk of copying between classmates. The complete list of activities of this stage, together with the knowledge that the students must acquire or apply, are shown in Table 11.

Table 11. Laboratory practices about server's learning topics: S = Server Programming, D = Database, F = Files, A = AJAX, I = Improved Front-End.

No.	Practices	Description	S	D	F	A	I
1	Login	Authentication mechanisms: database access and JavaScript validation. At least, 2 different user profiles: (a) Administrator and (b) Normal.	√	√			
2	Desktop	Depending to its profile: the user will have different level of access. Administrator: Users Manager Application; Normal user: all the other ones.	√	√			√
3	Users Manager	The Administrator can add, delete, update and list other users. Each user can update his own data.	√	√			√
4	Editor	Extend the Notes system of Table 10. Now the user can add, delete, update and list his own notes. Data persistence is required.	√	√	√	√	√
5	File Explorer	Create, delete, or rename plain-text files. The files must be stored on the web server.	√	√	√	√	√

6.2. Training Web Programming Skills by Developing Laboratory Practices

As mentioned in Section 5.2.2, grade must measure gradual personal growth, so we have divided the integrating project into smaller laboratory practices. We consider that the development of these practices conform the success of the project. Note that, before giving the specifications of each laboratory practice, the teacher must explain the topics that encompass them. We have classified the laboratory practices in three learning units (already described in Section 5.1).

Front-end (Table 9, No. 3–8): are practices focused on supporting the development and design of the project but also on clarifying topics such as HTML, CSS (respectively HTML5 and CSS3), JavaScript, and libraries. Table 10 describes each laboratory practice and the web technologies that the students are expected to learn and exercise:

- In the laboratory practice 1 the students learn the general structure of an HTML page together with basic tags to format fonts.
- In practice 2, through a **Login** html page design, the students enforce html tags like divs, inputs, forms, links, etc. and start using CSS.
- JavaScript concepts are included in practice 3, with a **Calculator** web page, students consolidate the different ways to include styles, CSS selectors, the use of ids and class; and start programming with JavaScript to do functionality to the web calculator.
- In Practice 4, designing a light version of an **Editor** system, students understand HTML DOM and JavaScript libraries; in this part, more experimented students could explore some front-end frameworks like Bootstrap, JQuery, etc.; data persistence is not requested yet.

Each of these laboratory practices is only an example; teachers following the idea of the laboratory practices could include other, similar ones. In addition, each of the laboratory practices must be connected to the selected operating system's look and feel.

Figure 4 illustrates an example of a Windows 95 operating system designed by one of the students. The left part of the figure shows the desktop and the options; the right part of the figure shows the calculator and the login interface.

Back-end (Table 9, No. 9–11 and 13–14): are practices to help the students understand concepts such as web server configuration: IP, setting a port number, session id, maximum amount of file uploads, etc; Database Management System (DBMS), FTP:

- The first laboratory practice, **Installing a Web Server**, consists in delivering a simple practice where students present a simple page executing it in the server side. Depending of the instructor experience, students might work with any particular Web Server technology, giving students freedom to explore different approaches; no matter the choice, it is important to take special care of the main configuration files that will be processed by the server.

- The next laboratory practice, **client side vs. server side**, comprehends which part of the script runs on the server and which part on the client; students must deliver a practice in two different computers: one being the web server and the other one being the client that will connect using a web browser.
- The third laboratory practice, **DBMS**, consists of the students understanding the separation and interaction between the three different entities, the User Interface, the Web Server and the Database Server; it involves knowing the required server programming sentences (instructions) to establish communication between them. CRUD operations in DBMS are left for the next unit. Taking into account that this activity is only 5.5 h long, security aspects are not considered because they are not part of this course but they are considered for more advanced courses.

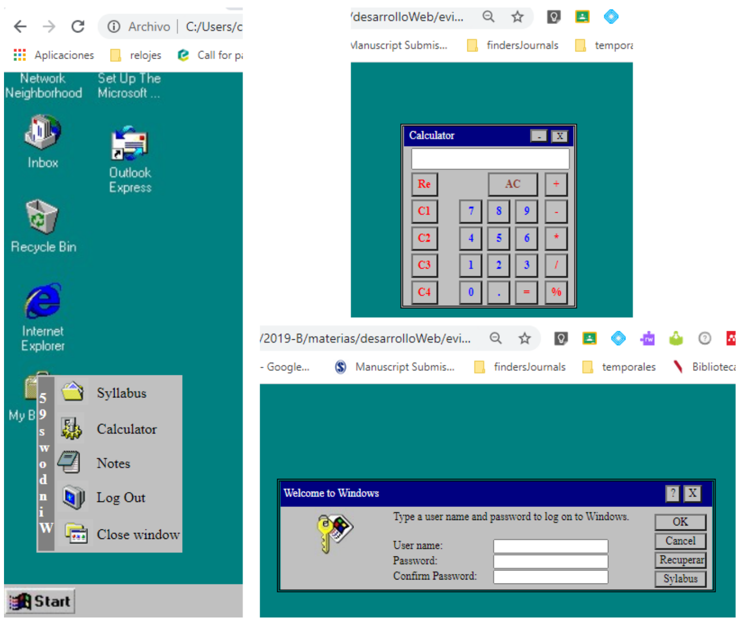

Figure 4. Front-end example: Windows 95 Operating System developed by one of the students.

We have identified that some students without a previous background in network topics have some difficulties in understanding the client and server communication paradigm. We recommend this topic to be studied previously. The list of activities of these practices, together with the knowledge that the students must acquire or apply, are shown in Table 12.

Table 12. Back-end laboratory practices and learning topics: W=Web Server Configuration, H = HTTP, C&S = Differentiate between client and server Programming, D = DBMS.

No.	Practices	Description	W	H	C&S	D
1	Web Server	Configuring a web server: IP, port number, session ids, upload size.	√	√		
2	Client Server	Web programming illustrating in what computer is running the front-end and in which the back-end.	√	√	√	
3	DBMS	A database server installation a doing a connection from a web programming.			√	√

Client–server interaction (Table 9, No. 15–20): in this part of the course, the students must have already achieved experience, at least a little, and knowledge about Front and Back End concepts. He/she must be prepared to incorporate a mixture of technologies to close his/her integrating project. At this point, the teacher will choose a server programming language to teach this paradigm. Years ago, we used ASP and JSP; recently we have turned to PHP or JavaScript server with Node.js. We recommend starting with PHP as a more didactic choice, as we have identified that students with no experience in the server–client paradigm get it faster. We have also noticed that the way JavaScript is processing in the server side gets the students confused with the functionality of the client side. Regardless of the choice, we have proposed practices incorporated in this unit.

Table 11 abstracts the following:

- **Cookies and database connection**: the practice consists in implementing a simple authentication mechanisms (see **Login** and **Desktop** in Table 11) by accessing a database and decide what type of user has been logged (Maiorana, [26], suggests a simple, useful version). In this practice, the students perfect their knowledge of the different instructions that can be programmed over the server side. They also improve their perspective knowledge of the client side (JavaScript) by avoiding data latency doing corresponding validations. It is suggested to apply cookie concepts.
- **CRUD operations in DBMS**: implement the **User Manager** system of Table 11, where it is possible to add, delete, update, and list user's personal data.
- **Files and Session variables**: it consists in implementing a text editor system (see **Editor** in Table 11) where each user can create, read, and update his own files. At least, with the following functional attributes: font family, font size, and the following commands: "do" and "undo" options. Files must be stored in the server side.
- **AJAX**: implement the **File explorer** described on Table 11, where each user can list and delete his own files in real time. It must be possible to open the listed files with the **notepad**. It must contain a file searcher using asynchronous communication, in other words, while the user types a key its file coincidences should be appearing.

Figure 5 illustrates front-end and back-end of various operating systems implemented by students. The left part shows a simple login in ubuntu using the HTTP protocol (see the URL); the right part at the top shows an editor system (notes) under macOS; the right part at the bottom shows a simple file explorer using AJAX in Windows XP. Note that the figure illustrates different user experience design. Each of them represents an individual effort by students.

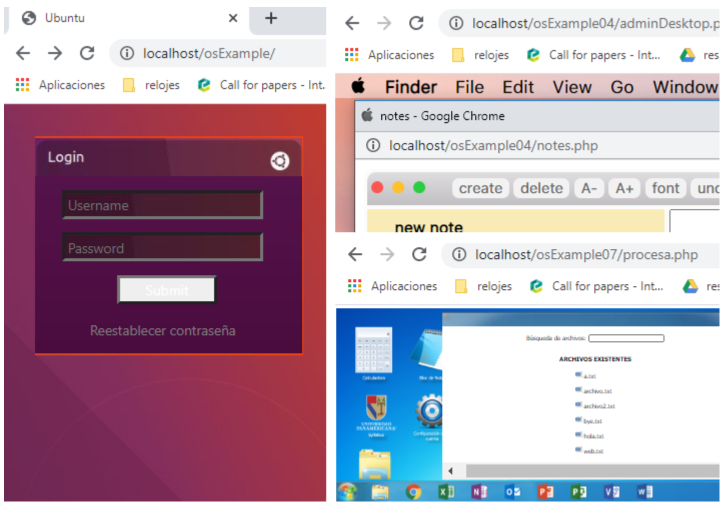

Figure 5. Remote Desktops developed by different students following already consolidated Operating Systems (Ubuntu, MacOS, and Ms Windows).

Finally, a reflection obtained from this last class taught remotely is that it helped the students to better understand the client–server concepts. For the laboratory practices, some students had to install the web and database servers on their own computers, and in other cases they did so by installing them on remote computers at the University.

6.3. Knowledge Evaluation

There are two exams being applied: midterm (Table 9 No. 12) and final (Table 9 No. 21). The midterm exam is split into two parts: (i) concepts and (ii) small problems. The concepts exam will include general and basic aspects about Front-end. Notice that this exam is not about the perfect memorization of HTML tags, CSS attributes, or JavaScript instructions but more about their general functionality. The second part consists in solving small application problems through the JavaScript language and HTML DOM.

The proposed final exam is also split into the same two parts: (i) concepts and (ii) a small problem. The concepts part has the objective of demonstrating the understanding on the client/server architecture, HTTP protocol, configuration ports, session variables, cookies, and AJAX. The problem part consists in solving a small problem using server communication and programming.

Lastly, the small problem in both exams could be as easy as developing a web page that prints the sum of two numbers or calculating the number of days between two dates. It is important that the student focus on the application of the studied technologies, more than in the difficulty of the problem presented.

7. Discussion and Conclusions

Web programming is becoming an indispensable requirement in software development, especially for the new generation of entrepreneurs and innovators, who want to have a formation focused on having a full-stack development, more than just using existing platforms. These new students' profiles want to employ themselves with their own ideas instead of just being employees. It is no longer enough to use traditional applications and visit web pages daily; the real challenge in the future is to create more innovative web applications rather than retrying traditional ones.

We believe that our methodology is an innovative teaching strategy for web programming courses. According to [45], an innovative teaching method is any communication method used to serve the purpose of teachers without destroying the objectives of the learning; based on curriculum requirement and students' needs. The described methodology has been continuously adapted and implemented (Table 1) because we have taken into account students' feedback, which has been used to consolidate the activities and general principles of the methodology. Questionnaires given to students have been used to polish the activities, like the ones shown in Tables 5 and 6.

The research questions presented in the Introduction are answered as follows:

(a) A web programming engineering-level course has been proposed in Section 5, the proposal includes a teaching/learning methodology based on PBL in order to build a common application to all students such as a Remote Desktop but having a different operating system design for each student (as reported in Table 3) in order to avoid plagiarism as discussed in Section 4.2.

(b) Every student can follow his/her own tempo when making their own Remote Desktop project as explained in Section 3.3, going to the depth they consider is required, as long as they complete the required tasks (as exemplified in Sections 6.1 and 6.2). It can be a real challenge to keep up with some details on the interfaces, but those details can be skipped without losing the general thread of the course. This has both the benefits of personalizing the learning process and also making it harder for the students to copy the work of their companions (the risk is lower with smaller groups as illustrated in Table 3).

(c) Table 2 shows how despite the years and the different technologies being introduced year after year, the methodology is still in force. Our methodology can be adapted and updated accordingly to the newest web tools that can and will be appearing, without being outdated.

We have also given some suggestions to colleges and teachers. The suggestions were specified to each of the colleges in their curricular plan that have already implemented the methodology. These can be applied to other curricula around the world that are in similar circumstances. Another suggestion consists in that teachers who are looking for a teaching/learning strategy in unique courses involving Web Application Development might use this research. This strategy is appropriate as long as the teacher faces students with previous knowledge of structured and object-oriented programming and, it would be formidable with students with previous knowledge of databases and network communication.

Author Contributions: Conceptualization, J.C.L.-P.; data curation, J.C.L.-P.; formal analysis, J.C.L.-P. and M.A.-R.; funding acquisition, J.C.L.-P., A.M.-S., M.A.-R., C.D.-V.-S.; investigation, J.C.L.-P.; methodology, J.C.L.-P.; project administration, J.C.L.-P. and A.M.-S.; resources, J.C.L.-P. and C.D.-V.-S.; supervision, J.C.L.-P., A.M.-S. and M.A.-R.; validation, J.C.L.-P. and A.M.-S.; writing—J.C.L.-P. and M.A.-R.; writing—review and editing, J.C.L.-P., A.M.-S. and C.D.-V.-S.; All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Answers to the surveys applied can be available at https://git.io/JegmF accessed on 1 June 2021, materials of one of our courses can be available at https://git.io/J8at9 accessed on 1 June 2021.

Acknowledgments: We thank for his initial comments to improve the paper to Luis Alberto Morales Rosales affiliated at Facultad de Ingeniería Civil, CONACyT-Universidad Michoacana de San Nicolás de Hidalgo, Morelia, Michoacana, lamorales@conacyt.mx. We also thank the anonymous reviewers for their useful suggestions have complemented and strengthened enormously our paper.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Cabada, R.Z.; Estrada, M.L.B.; Hernández, F.G.; Bustillos, R.O.; Reyes-García, C.A. An affective and Web 3.0-based learning environment for a programming language. *Telemat. Inform.* **2018**, *35*, 611–628. [CrossRef]
2. Wasserman, A.I. How the Internet transformed the software industry. *J. Internet Serv. Appl.* **2011**, *2*, 11–22. [CrossRef]
3. Milková, E.; Ambrožova, P. Internet Use and Abuse: Connection with Internet Addiction. *J. Effic. Responsib. Educ. Sci.* **2018**, *11*, 22–28. [CrossRef]
4. Marinoni, G.; Van't Land, H.; Jensen, T. The impact of Covid-19 on higher education around the world. *IAU Glob. Surv. Rep.* **2020**. Available online: https://www.iau-aiu.net/IMG/pdf/iau_covid19_and_he_survey_report_final_may_2020.pdf (accessed on 20 July 2021)
5. Milne, I.; Rowe, G. Difficulties in Learning and Teaching Programming—Views of Students and Tutors. *Educ. Inf. Technol.* **2002**, *7*, 55–66. [CrossRef]
6. Piteira, M.; Costa, C. Learning Computer Programming: Study of Difficulties in Learning Programming. In Proceedings of the 2013 International Conference on Information Systems and Design of Communication, ISDOC'13, Lisboa, Portugal, 11–12 July 2013; ACM: New York, NY, USA, 2013; pp. 75–80. [CrossRef]
7. Kalelioğlu, F. A new way of teaching programming skills to K-12 students: Code.org. *Comput. Hum. Behav.* **2015**, *52*, 200–210. [CrossRef]
8. Román-González, M.; Pérez-González, J.C.; Jiménez-Fernández, C. Which cognitive abilities underlie computational thinking? Criterion validity of the Computational Thinking Test. *Comput. Hum. Behav.* **2017**, *72*, 678–691. [CrossRef]
9. Durak, H.Y.; Saritepeci, M. Analysis of the relation between computational thinking skills and various variables with the structural equation model. *Comput. Educ.* **2018**, *116*, 191–202. [CrossRef]
10. Figueiredo, J.; García-Peñalvo, F.J. Building Skills in Introductory Programming. In Proceedings of the Sixth International Conference on Technological Ecosystems for Enhancing Multiculturality, TEEM'18, Salamanca, Spain, 24–26 October 2018; ACM: New York, NY, USA, 2018; pp. 46–50. [CrossRef]

11. Nouri, J.; Zhang, L.; Mannila, L.; Norén, E. Development of computational thinking, digital competence and 21st century skills when learning programming in K-9. *Educ. Inq.* **2019**, *11*, 1–17. [CrossRef]
12. Wang, Y.D.; Zahadat, N. Teaching Web Development in the Web 2.0 Era. In Proceedings of the 10th ACM Conference on SIG-information Technology Education, SIGITE '09, Fairfax, VA, USA, 22–24 October 2009; ACM: New York, NY, USA, 2009; pp. 80–86. [CrossRef]
13. Liu, Y.; Phelps, G. Challenges and Professional Tools Used when Teaching Web Programming. *J. Comput. Sci. Coll.* **2011**, *26*, 116–121.
14. Xinogalos, S.; Kaskalis, T.H. The Challenges of Teaching Web Programming—Literature Review and Proposed Guidelines. In Proceedings of the WEBIST 2012—8th International Conference on Web Information Systems and Technologies, Porto, Portugal, 18–21 April 2012; WEBIST: Porto, Portugal, 2012; pp. 207–212.
15. Douce, C. Teaching web technologies: Understanding the tutor's perspective. *Open Learn.* **2019**, *34*, 78–88. [CrossRef]
16. Rosenbloom, A.; Zhang, L.Y. A 12 Week Full Stack Web Course in 2017. In Proceedings of the 2017 ACM Conference on Innovation and Technology in Computer Science Education, ITiCSE '17, Bologna, Italy, 3–5 July 2017; ACM: New York, NY, USA, 2017; pp. 86–87. [CrossRef]
17. Bosse, Y.; Gerosa, M.A. Why is Programming so Difficult to Learn? Patterns of Difficulties Related to Programming Learning Mid-Stage. *SIGSOFT Softw. Eng. Notes* **2017**, *41*, 1–6. [CrossRef]
18. Malik, S.; Shakir, M.; Eldow, A.; Ashfaque, M.W. Promoting Algorithmic Thinking in an Introductory Programming Course. *Int. J. Emerg. Technol. Learn. (iJET)* **2019**, *14*, 84–94. [CrossRef]
19. Mehmood, E.; Abid, A.; Farooq, M.S.; Nawaz, N.A. Curriculum, Teaching and Learning, and Assessments for Introductory Programming Course. *IEEE Access* **2020**, *8*, 125961–125981. [CrossRef]
20. Guo, D.; Koufakou, A. A comprehensive and hands-on undergraduate course on cloud computing. In Proceedings of the ASEE Southeastern Section Conference, Daytona Beach, FL, USA, 4–6 March 2018; American Society for Engineering Education: Washington, DC, USA, 2018; pp. 1–6
21. Maiorana, F. Extending the Database Curriculum: From Design Principles to Web and Mobile Programming. In *Computer Supported Education*; Zvacek, S., Restivo, M.T., Uhomoibhi, J., Helfert, M., Eds.; Springer International Publishing: Cham, Switzerland, 2015; pp. 258–271.
22. Connolly, R. Facing Backwards While Stumbling Forwards: The Future of Teaching Web Development. In Proceedings of the 50th ACM Technical Symposium on Computer Science Education, SIGCSE '19: Minneapolis, MN, USA, 27 February–2 March 2019; Association for Computing Machinery: New York, NY, USA, 2019; pp. 518–523. [CrossRef]
23. Adams, D.R. Integration early: A new approach to teaching web application development. *J. Comput. Sci. Coll.* **2007**, *23*, 97–104. [CrossRef]
24. Kar, S.; Islam, M.M.; Rahaman, M. State-of-the-Art Reformation of Web Programming Course Curriculum in Digital Bangladesh. *Int. J. Adv. Comput. Sci. Appl.* **2020**, *11*, 193–201. [CrossRef]
25. Connolly, R.W. Awakening Rip Van Winkle: Modernizing the Computer Science Web Curriculum. In Proceedings of the 16th Annual Joint Conference on Innovation and Technology in Computer Science Education, ITiCSE '11, Darmstadt, Germany, 27–29 June 2011; ACM: New York, NY, USA, 2011; pp. 18–22. [CrossRef]
26. Maiorana, F. Teaching Web Programming—An Approach Rooted in Database Principles. In *Proceedings of the 6th International Conference on Computer Supported Education—Volume 2*; CSEDU; INSTICC; SciTePress: Barcelona, Spain, 2014; pp. 49–56. [CrossRef]
27. Shropshire, J.; Landry, J.P.; Presley, S.S. Towards a consensus definition of full-stack development. In Proceedings of the Southern Association for Information Systems Conference, St. Augustine, IL, USA, 23 March 2018; pp. 1–6
28. Fioravanti, M.L.; Sena, B.; Paschoal, L.N.; Silva, L.R.; Allian, A.P.; Nakagawa, E.Y.; Souza, S.R.; Isotani, S.; Barbosa, E.F. Integrating Project Based Learning and Project Management for Software Engineering Teaching: An Experience Report. In Proceedings of the 49th ACM Technical Symposium on Computer Science Education, SIGCSE '18, Baltimore, MD, USA, 21–24 February 2018; Association for Computing Machinery: New York, NY, USA, 2018; pp. 806–811. [CrossRef]
29. García-Peñalvo, F.J. Map of trends in educational innovation. *Educ. Knowl. Soc.* **2015**, *16*, 6–23. [CrossRef]
30. Mills, J.E.; Treagust, D.F. Engineering education—Is problem-based or project-based learning the answer. *Australas. J. Eng. Educ.* **2003**, *3*, 2–16.
31. López-Pimentel, J.C.; Sánchez G.J.; González L.A.; Valles L.I.; Ríos T.A.; Gutiérrez G.M.; Suárez R.F. Enseñanza de Tópicos en Programación Web usando Resultados de Aprendizaje Basados en Evidencias—Teaching Topics in Web Programming using Evidence-Based Learning Results. In *Programación Matemática y Software*; Universidad Autónoma del Estado de Morelos: Cuernavaca, Mexico, 2018; pp. 50–62
32. Silberschatz, A. *Operating System Concepts*; Wiley: Hoboken, NJ, USA, 2018.
33. Wang, X. Design, Develop and Teach the Second Web Programming Course in Computer Science Curriculum. *J. Comput. Sci. Coll.* **2014**, *29*, 52–59. [CrossRef]
34. Harriger, A.R.; Woods, D.M. A Structured Approach to Teaching Web Development. In Proceedings of the World Conference on the WWW and Internet WebNet, Orlando, FL, USA, 23–27 October 2001; SciTePress: Barcelona, Spain, 2001; pp. 23–27
35. Margaret, R.; Vinod, A.M.; Tejonidhi, M.R. A Practical Approach to Teach Web Programming Course. *J. Eng. Educ. Transform.* **2016**, *30*, 99–102. [CrossRef]

36. Muibi, H.; Dorn, B.; Park, T. Teacher Perspectives on Web Design Instruction. In Proceedings of the 2015 ACM Conference on Innovation and Technology in Computer Science Education, ITiCSE '15, Vilnius, Lithuania, 4–8 July 2015; ACM: New York, NY, USA, 2015; pp. 231–236. [CrossRef]
37. Connolly, R. Criticizing and Modernizing Computing Curriculum: The Case of the Web and the Social Issues Courses. In Proceedings of the Seventeenth Western Canadian Conference on Computing Education, WCCCE '12, Vancouver, BC, Canada, 4–5 May 2012; ACM: New York, NY, USA, 2012; pp. 52–56. [CrossRef]
38. Dugan, R.F., Jr. A Single Semester Web Programming Course Model. *J. Comput. Sci. Coll.* **2013**, *29*, 26–34.
39. Deek, F.P.; Hiltz, S.R.; Kimmel, H.; Rotter, N. Cognitive Assessment of Students' Problem Solving and Program Development Skills. *J. Eng. Educ.* **1999**, *88*, 317–326. [CrossRef]
40. Scherer, R.; Siddiq, F.; Sánchez Viveros, B. The cognitive benefits of learning computer programming: A meta-analysis of transfer effects. *J. Educ. Psychol.* **2019**, *111*, 764–792. [CrossRef]
41. Anderson, L.W.; Krathwohl, D.R. (Eds.) *A Taxonomy for Learning, Teaching, and Assessing. A Revision of Bloom's Taxonomy of Educational Objectives*, 2nd ed.; Allyn & Bacon: New York, NY, USA, 2001.
42. Walraven, A.; Brand-Gruwel, S.; Boshuizen, H.P. How students evaluate information and sources when searching the World Wide Web for information. *Comput. Educ.* **2009**, *52*, 234–246. [CrossRef]
43. Tang, A.; Aleti, A.; Burge, J.; van Vliet, H. What makes software design effective? *Des. Stud.* **2010**, *31*, 614–640. [CrossRef]
44. Hijazi, S. Using Portable Technology to Teach Web Programming and Database Classes. In Proceedings of the 46th Annual Conference, Myrtle Beach, SC, USA, 9–13 June 2013; pp. 34–50
45. Walekar, M.M.; Shinde, M.D. Innovative Teaching Methodologies in Applied Sciences. *Natl. J. Comput. Appl. Sci.* **2018**, *1*, 29–35.