REMOTE LEARNING EXPERIENCES WITH INTRODUCTORY WEB PROGRAMMING COURSE

M. Mikac, R. Logožar, M. Horvatić

University North (CROATIA)

Abstract

After years of teaching an introductory web development course for our undergraduate bachelor students in electrical engineering, we had to offer the same course as a remote e-learning course during the COVID-19 crisis. Our intention was to cover all or at least most of the topics included in the standard course. It was quite important for us to carry out both the lectures and practical computer labs. The online lectures can be mostly performed as one-way, less-interactive remote learning, but for computer labs, we had to choose a solution allowing two-way interaction, including the procedure allowing students to interactively present their current work. Additionally, the exams were also organized remotely. Our approach used during a remote e-learning form of the course is described. As obligated by our university, all the lectures were scheduled and performed live, as synchronous lessons. Additionally, asynchronous recordings of those lectures were offered to students via YouTube video services. Computer labs were organized similarly, but with some specifics - basically, introduction lessons for each lab were presented as synchronous events being offline available as video recordings. But, the most important part of the labs - interactive, two-way sessions had to be organized differently. That is where we point out all the benefits of using freely available tools and technologies on student premises. Two-way interaction and its importance for remotely performed labs, as well as exams, is additionally stressed.

As part of the paper, the student results are commented and our subjective opinion on student activities and interests, and overall course performances are given. In our opinion, we can conclude that remote learning can be used for these kinds of practical, skill-oriented, courses.

Keywords: web programming, web app, web applications, online e-teaching, remote learning, colleague course, COVID-19, teaching experience.

1 INTRODUCTION

During the intensive COVID-19 pandemic crisis, the lectures at our university were dominantly performed as online, remote e-learning courses – the directives of our university leadership stated that, during the pandemics, the lectures and exercises should be dominantly in an online form. Exceptionally, some classes and particularly the laboratory exercises were allowed onsite, but with certain limitations and adapted to epidemiology guidelines. Of course, it was our intention that in this new, *e-form*, the course retain as much of the contents of its normal form as possible, and to achieve most of the educational goals. While the practical labs requiring specialized equipment usually not available to the students cannot be performed remotely with the same coverage as on-site, software-based labs should be (or at least we anticipate it that way) much easier to replace with e-learning solutions without noticeable impact to the lecture quality. Some practical, skill-based lectures, like the one described in this paper – web programming course, or other programming courses such as mobile app developments course [1], were successfully held remotely [2].

This paper presents our remote learning experiences with above-mentioned web programming course. It will describe the approach used in online teaching of both the lectures and programming exercises in computer labs, emphasizing two-way communication during the labs (and exams). Essentially, it is based on our approach of combining synchronous lectures and their asynchronous recordings described in [3]. That approach was extended with two-way communication, on-demand remote screen sharing and similar features of the software tools freely available during COVID-19 outbreak.

The paper is organized as follows: after this section, general information about the observed course is given, with remarks related to the remote e-learning model used. Next section gives an overview of the technical details and requirements for the online form od the course. Section four discusses the fulfilment of the learning outcomes, student success and the score on their final exams, in pre-COVID-19 during COVID-19 academic years. The last section concludes the paper with several remarks regarding the achieved educational goals - it is concluded that most of the students successfully managed to keep

track of all the course topics and that obtained results (exams and resulting practical apps) were in fact not so different from those obtained in pre-COVID-19 conditions.

2 DESCRIPTION OF THE WEB PROGRAMMING COURSE

The web programming course, in its standard face-to-face form, has been taught on the study of electrical engineering at the University North, Croatia, since the academic year 2015/16. It is an elective course that is offered to our sophomore students (second year electrical engineering undergraduate bachelors, with the three-year bachelor degree education), The course is skill- and example-based, with intention to introduce and teach our students the basics of web programming. It is expected that students who pass the exam should be able to develop simple web applications, both the client- and server sided, and use that knowledge in solving their future professional problems. The course is credited with 3 ECTS points and is based on 30 hours of lecture lessons and 15 hours of computer labs. Most of the lectures and all labs are performed as practical as possible, using a standardized development environment. The students are obligated to attend all computer labs, while the attendance to the lectures is optional.

Currently, the suggested development environment includes Visual Studio Code (VSC) [4], any available web browser with integrated developer tools (Google Chrome, Microsoft Edge, Mozilla Firefox, or other) and XAMPP [5] (or WAMP [6]) installation package for Windows operating system. Computers on our university premises are usually equipped with Windows OS, so that this platform is considered as default. We also encourage out students to check alternative solutions and environments. Those familiar with other operating systems were instructed to use integrated servers or XAMPP alternatives.

There are no special equipment requirements for the course and suggested environment can be easily installed and used even on older computers (in contrast to more specialized and complex mobile application development environment described in [1]). This turned out to be a mitigating factor when facing the COVID-19 crisis and the e-learning requests.

2.1 Education prerequisites

In the first year of their education, our students have three mandatory courses that may be considered prerequisites or at least related to this course. The first one is a classic basic-programming course, with C/C++ being the languages of choice. In a short comment to it, we must emphasize that despite the expectation that “STEM students” should be rather familiar with basic programming after their high school and other secondary-level education, the experience on our institution and with our students shows quite dispersed or low programing-related knowledge and skills [7]. Thus, we had to start that course with elementary introduction to programming, followed by details of C/C++ basics – variables, flow-control, arrays and functions. Languages used in the web programming course – JavaScript for the client side and PHP for server-side development – are therefore initially introduced with that in mind, without in-depth coverage of their specifics. Instead, we point out the differences and similarities to the C/C++, and check them on examples during the programming exercises.

Another partially related course introduces TCP/IP computer networks, including the HTTP as one of the most used application layer protocols. It is expected that the students remember the basic idea behind it, particularly the part related to the client-server request-response communication. Since the HTTP represents the basic protocol used for communication and data-exchange in web applications, the knowledge of it is considered a prerequisite.

In their second year, prior or in the same semester as is the web programming course, our students listen to another mandatory computer-science course – Databases and SQL. Because the subject is an important part of the server-side related programming, it is an important prerequisite, too.

2.2 Course topics

The most important topics of our web programming course are shown in Fig. 1. After introduction lesson, which covers the basic idea of web pages and web application, and presents the client/server concept, the course is divided in two main parts: 1) the client-side related content definition and programming, and 2) the server-side programming. The figure shows nine basic topic blocks, which relate to nine planned lecture lessons and five computer labs.

The client-side part includes topics covering basics of HTML (Hypertext Markup Language), CSS (Cascading Style Sheets) and JavaScript programming language. The HTML is briefly introduced as the markup language used for defining the structure and the content of both webpages and web

applications. CSS is also briefly covered, with enough details allowing students to stylize their web pages or applications. The basics of JavaScript are explained and related to browsers' document object model (DOM), in order to allow students to start client-side programming. As an important topic, the HTML form is explained, including the basic client-side processing of the form. In addition to plain CSS, some often used CSS frameworks such as Bootstrap [8] and W3.CSS [9] (W3Schools CSS framework, as an alternative to Bootstrap) are introduced prior starting with server-side topics and additionally illustrated by examples in some of computer lab assignments.

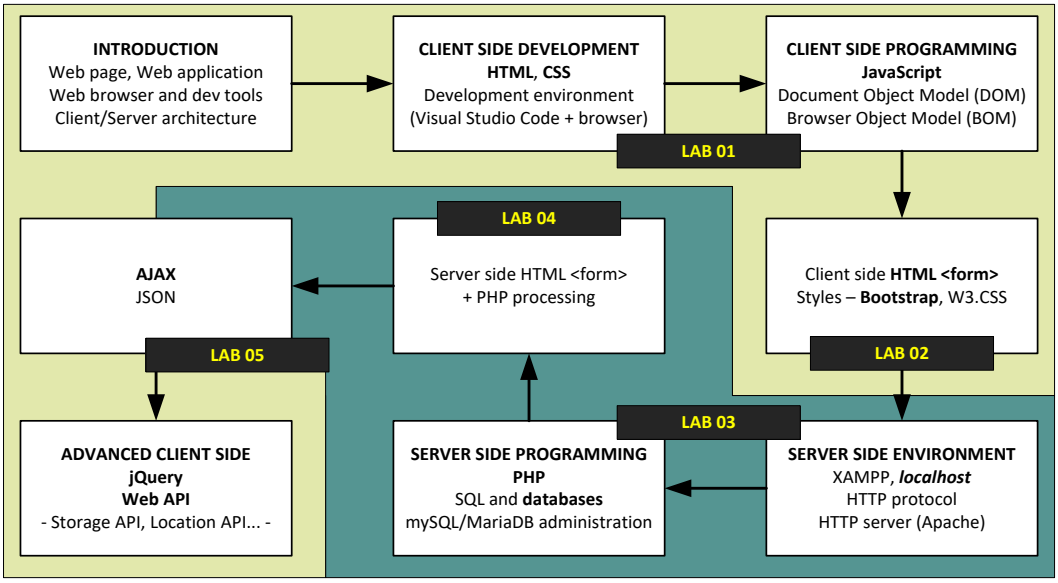


Figure 1. Visualization of the course topics

The server-side part includes the initial lesson introducing the server side environment installation - it relates to the *localhost* installation using XAMPP or WAMP, followed by introduction of the used subsystems: *Apache* HTTP server [10] accompanied with *MariaDB* [11] or *MySQL* [12] database server. Database administration tools such as web-based *phpMyAdmin* [13] or the standalone application *MySQL Workbench* [14], are briefly explained and suggested as default tools. Server related programming is done using the PHP. Due to the time limits and quite wide topic coverage, other server-side languages were not considered, together with some popular PHP-based frameworks. The most important topics with PHP included the HTML form server-side processing, using the classic HTTP requests from a web browser.

The final lessons explain very important topics that can be used in conjunction with single page application (SPA) – the HTTP server communication approach with AJAX (Asynchronous JavaScript and XML), which uses the standard JavaScript on the client-side. Related subjects such as JSON (JavaScript Object Notation) are also briefly presented. The last lesson introduces few advanced client-side topics: current standard web API's like Storage API or Location API, which are supported in most modern web browsers. The JavaScript library, jQuery, which is lately somewhat less popular, is only lightly touched in order to make the students aware of it and familiar with its specifics.

The programming exercise were organized as five 3-hour "lab terms", covering the topics presented in the lectures (Fig.1). All students had to complete and document the standard lab assignments. In addition to that, the advanced and ambitious students could have their special individually tailored assignments.

3 E-LEARNING MODEL DURING THE COVID-19 CRISIS

The course had to be rearranged in order to offer adequate e-learning for our students during the COVID-19 pandemic (in the academic years 2019/20 and 2020/21). While in 2019/20, a few lessons before March 2020 were given in standard form, on-site, in 2020/21 only one lecture was on-site (little less than 4 hours out of 30, satisfying all the epidemiology guidelines). The rest of the course was provided online, remotely, performed live as synchronous e-learning, following the official timetable schedule.

At the beginning of the COVID-19 crisis, our university defined some ground rules and decisions – in middle of March 2020, it was decided that all the lectures had to be provided remotely. At first, it was planned for only two weeks, but the scheme was, with some minor changes in the procedure, prolonged till the end of the classes in the academic year 2019/20. Precise numbers given in result section will provide a-posteriori information about implementation of our remote e-learning model for this course, without interfering with the formal statements and rules defined by our institution or other authorities.

Initially, the most of the teaching stuff was not familiar with real-time videoconferencing and remote learning tools, most of us were also completely unexperienced with recording sessions and lessons and any other e-learning related stuff. So, the first thing to catch up was to decide how to perform remote lectures and exercises for the course. There were "rumors" about some specialized tools such as *Zoom*, *Meet* and a few others. Luckily, our institution provided some basic information and the course lecturer (the first author of this paper) decided to use Google *Meet* [15] as remote collaboration and video conferencing platform for this course. As hinted in the introduction, the basic idea was to perform lectures as scheduled in synchronous e-classes, in which Meet served as videoconferencing platform allowing the lecturer to share his computer screen with *Powerpoint* presentation or any other content. That was called synchronous e-learning (lessons as scheduled for real on-site lessons). Additional challenge was, again required by the institution, to provide recordings of that scheduled sessions for something called asynchronous e-learning. Of course, the simplest and less complicated way to do that would be simply publishing recorded *Meet* sessions – this is a straightforward procedure, after having the formal approval from all the participants. But, as described in [3], the first author gave a try to little different, more complex approach – resulting in better quality of the recordings (the full HD video, accompanying camera recording of the lecturer). Furthermore, in this approach, there are no GDPR (General Data Protection Regulation) problems due to the visible student details, as it happens in standard Meet recordings. Finally, that kind of approach led to providing fully publicly available list of video lectures on first author's YouTube channel [16].

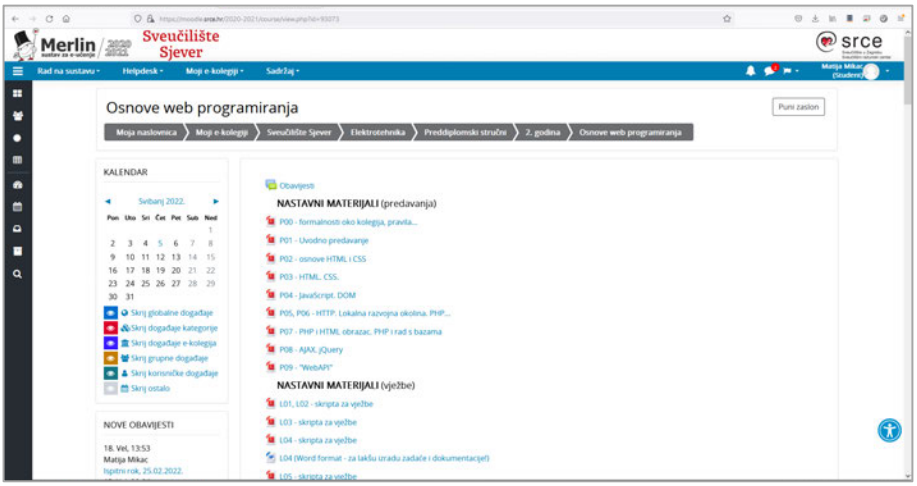


Figure 2. The web programming course LMS offering offline materials

As described in section 2, the installation and setup of the required development environment should not present a problem for the students. There are no special equipment requirements so that it was expected that students can simply prepare the environment on their own. The accompanying course materials – consisting of the PDF documents of the lectures and programming exercises assignments – were already prepared from before. Those were available to our students via our learning management system (LMS) called *Merlin* [17] – it is the Croatian "fork" of globally famous *Moodle* [18]. Standard user interface of our LMS, related to the web programming course is given in Fig. 2. The only addendum to it was new document section that was used to point to published video materials on *YouTube*, as well as inclusion of *Meet* links to scheduled e-learning synchronous sessions.

There were no course content changes and all the topics were included in remote learning sessions.

3.1 Software used and some technical details

In addition to the standard LMS, many software tools were checked and, after certain adaptation and analysis, selected as necessary to provide proper e-learning model and content. Details explaining the process of tool selection can be found in authors paper [3], so that only short information is given here.

As already mentioned, we have chosen Google *Meet* as videoconferencing, remote collaboration and online presentation system – it was used for both lectures and programming exercises, as well as for exams that were conducted remotely using the same online platform, but with extended usage of two-way communication and content sharing. One of the main reasons for selecting *Meet* was its integration with Google *GSuite* services, which are included in student and teaching staff e-mail accounts.

Software used to present prepared lectures content was standard *Powerpoint* presentation tool – sometimes, integrated functionalities allowing drawing and handwriting using the electronic pen or touchscreen were used. In some cases, external drawing application such as Microsoft *Whiteboard* or *OpenBoard* were used, with or without before-prepared overlay materials (Fig. 3). For presentation of the programming exercises tasks and computer labs content, standard PDF materials were used. For the practical part of the labs we have used VSC and all a few other software tools.

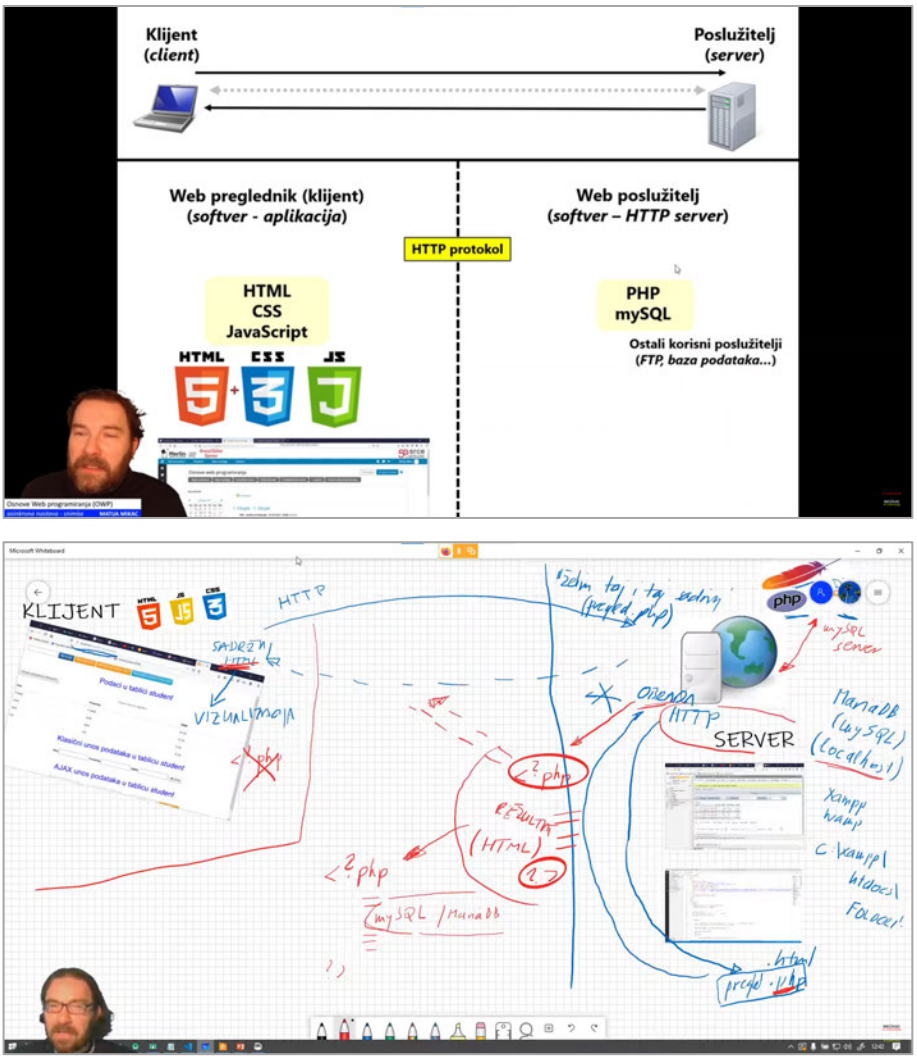


Figure 3. Recorded lectures showing usage of PowerPoint and WhiteBoard tool

All the remotely held sessions were recorded locally, using *OBS Studio* [19]. It is a very popular video recording freeware. Recorded materials were published to YouTube channel [16] and listed in LMS as additional content (for asynchronous e-learning). Prior to publishing, the videos were usually transcoded using *HandBrake* [20], making them more size-efficient. Most of the materials were published "as-is", without any post-production. Rarely, we used also *Microsoft Video Editor* or *ShotCut* [20].

As for the hardware requirements – standard computer equipment was used. Two webcams were used – one, of lower quality, integrated on notebook computer, was used for the real-time meeting via *Meet*. Another was used for recording teachers face and putting it as overlaid thumbnail to the video recorded in OBS. In order to get better thumbnail effect, some tweaking with the green-screen and light setup had to be done in order to remove thumbnail background. It is far from professional approach, but it was satisfying and definitely better than the direct Meet recordings.

3.2 Synchronous learning

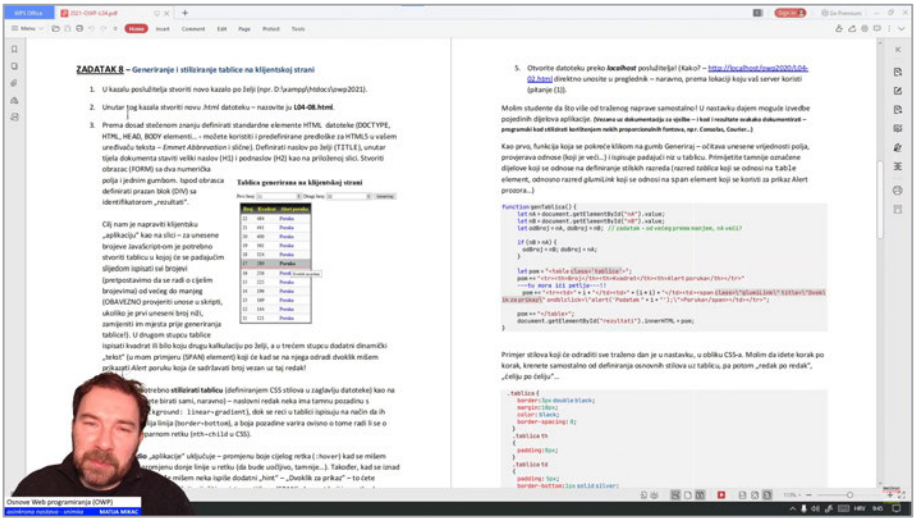
The complete learning process, both the lectures and exercises/labs, were scheduled according the official timetable. The intention was to keep students occupied as in standard learning process, since it was expected that non-scheduled lessons would not attract students, especially not in this special and unusual lockdown situation. That is why synchronous learning was selected as our main approach.

3.2.1 Synchronous lectures with asynchronous recordings

As already stated, video recordings of the synchronous lectures were published making it available as asynchronous e-learning content. It is important to say that asynchronous recordings included only one-way part of the synchronous lectures. The parts including discussions, student questions and other interactive two-way communication was excluded from recordings (partially due GDPR and our intention of not including students and their activities in publicly available materials). By assuring that, we could say that recorded teaching materials are solely content-related, since the topics were presented as in regular teaching approach.

3.2.2 Computer labs and synchronous learning

In contrast to the lectures, the online computer labs required greater reorganization and differ more from regular learning process. Depending on the totaln number of course participants, a few exercise groups are usually defined (15-20 students in each group, even though smaller groups may be better, space-limitations represent an issue). When providing the classes remotely, the lecturer decided to organize an introductory synchronous online class for each lab topic (total of 15 hours per student for computer labs are usually divided into 5 labs), a week or few days before regular scheduled term. The recorded session was also available for asynchronous learning, allowing all students to prepare for their scheduled term. Different than during lecture classes, lab related classes included analysis of tasks and assignments available in PDF documents, together with real examples presented directly from lecturer's screen – example screenshots of recorded lab classes depicted in Fig. 4 illustrate simple explanation of official document with assignments, as well as the usage of *Visual Studio Code* for programming the example given in accompanying document.



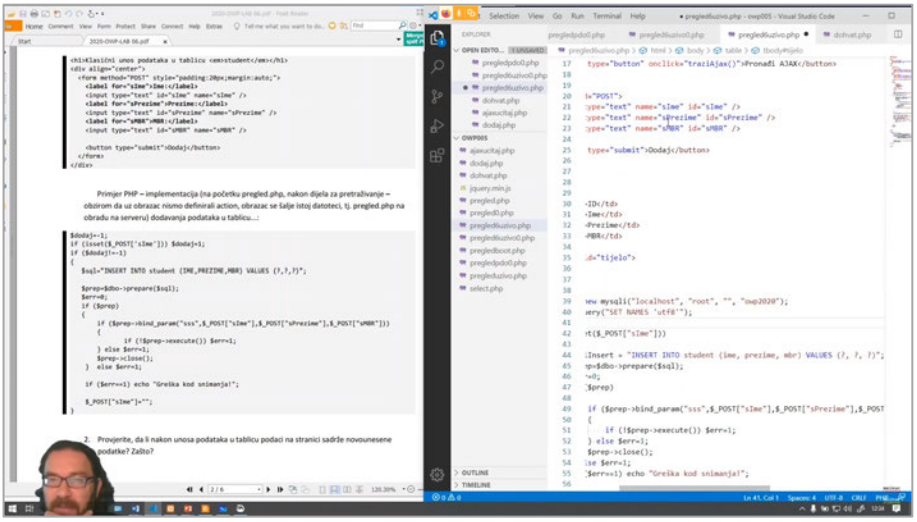


Figure 4. Recorded computer lab classes showing real-time programming in VSC

There were some differences in performing synchronous lab classes. Sometimes, all students were invited to participate a single online session. On the other hand, there were also classes where the students were divided into two or more groups. All sessions included practical work, with lecturer sharing his desktop screen, showing real-scenario development process, explaining usage of specialized tools, testing the applications etc. Student questions and discussions were allowed and it was lecturers intent to include as many students in two-way collaboration. That type of classes was not locally recorded, there were only *Meet* recordings which were not used for later publications – it was just a way of archiving and proving that the classes were held, including the ability to detect student's presence.

3.3 Two-way communication during remote learning sessions

Very important part of the remote e-learning is a two-way collaboration and live communication with students. Luckily, modern video conferencing tools such as *Meet* include that functionality from their initial version, with some improvements being add during the COVID-19 crisis. In short, each of the conference attendants can use both voice and video communication, as well as start sharing her screen. Of course, the teacher, as a moderator, has control mechanism to block or ignore certain participants.

The most important aspects of two-way communication during the synchronous sessions relates to programming exercises, or computer labs as we popularly call them. Another important use of such communication is during the tests or control of student activities. Of course, in order to be able to identify students, they were obligated to turn on their camera during two-way communication sessions.

3.3.1 Computer labs and two-way communication

Following the requirements specified in the PDF programming exercise materials, the students had to do their preparatory written assignments, answering several theoretical questions about each lab topic. The assignments were supposed to be uploaded to the LMS in due time for each programming exercise.

During the synchronous lab sessions, the students' work was additionally supervised, by oral questions about their homework assignments. That kind of supervision is good to instigate the activity of the class participants and to keep them aware of the necessity to prepare for the practical programming. Obviously, this required efficient two-way teacher-student communication. The same applied also to correcting the student's programming and project assignments.

3.3.2 Exams and tests

Besides the lectures and exercises, all the tests and exams in the two COVID-19 academic years were also given online. We have used our LMS platform to conduct written exams with various types of questions. We have created several sets of questions, were each set covered a part of course syllabus. Then the LMS created a test for each applicant by randomly choosing a question for him or her.

In order to enable us to check the student activities during the tests and exams, they had to turn on their web cameras. This was a basic but efficient way of supervising the participants. There are some sophisticated add-ons that can be integrated in LMS in order to increase the level of control of the

student computers (e.g. *Safe Exam Browser*), but we didn't use it in this course. The time limit for the online exams of this course were set to be quite short. We hope that this have both discouraged and disabled our students to cheat.

The oral exams were also organized online, usually in smaller groups, of 3 to 5 students. In a two-way one-on-one communication, we required from students to give direct answers to the questions that we asked them, expressing themselves both orally and in written form, as well as by showing their programming code.

3.3.3 Two-way communication and interaction drawbacks

The only drawback we experienced in our usage model was lack of remote control in *Meet* – without it, we actually cannot even talk of real two-way interaction, but just about two-way communication and one-way live presentation of the attendee screen. Presenting a screen during the conference is standard functionality, but it works only one-way. There is no way for teacher to take control of remote screen/desktop and, for example, correct or rewrite student's code in real-time.

There are many commercial and free tools that may allow remote control, but, at the time of performing this course, we did not find any that could directly integrate in Meet conference session. Even though Google offers *Chrome Remote Desktop* [22] which may be quite simple to integrate, it was not used neither analyzed by us during this course. It may be considered for future use, expanding the concept of two-way communication. But, of course, we would be happier if we won't need to use it, at least not due epidemics or other threatening scenarios.

4 STUDENT ACTIVITIES, OUTCOMES AND RESULTS

To give fully objective estimates of the student activities and results in the web-programming course during the COVID-years is a hard, if not an impossible task. We can say that most of our students — including those who did not pass the exam in the given academic year — were successful in fulfilling all the given requirements and solving their assignments. Before outlining the objective results, we summarize the student activities and the learning outcomes of this course.

4.1 Student activities

The student activities for the web programming course in regular on-site teaching include attending the lectures, attending the labs and documenting programming exercises, solving homework and given assignments. The presence on the lectures is not mandatory, but students are obligated to attend all computer labs! Homework and written assignments are sometimes evaluated and scored, while sometimes just used to "activate" students. Additionally, as part of continuous activity tracking and evaluation, written tests or quizzes are given to the students during the lectures and/or lab sessions.

Similar activities were expected during the remote learning – as pointed out, both lectures and labs were performed synchronously, allowing attendance tracking using *Meet*. Homework assignments and lab preparations were required and uploaded using Merlin LMS. Short test, quizzes and, finally, the exams, were taken in written form using LMS, while oral examination was conducted using two-way interaction and video conferencing. Both homework and tests were evaluated, allowing the best students to be exempt of the final written exam.

4.2 Learning outcomes

Based on topics covered with the course, the expected outcomes include certain level of coping with web programming and skills allowing students to be able to solely create (and even design or stylize) web pages and client-side web applications, and to develop programming logic implemented on both client and server side using JavaScript and plain PHP. Students who scored higher grades should be able to develop more complex applications and be familiar with AJAX, database programming and usage of web API's. Students not being able to manage basics of the web programming cannot get positive evaluation and should consider changing their elective course or repeat this course and try to increase their efforts with it.

4.3 Results. Student success

As our measure of the student success, we have outlined the pass rates and the average grades per year. These numbers were obtained from the official Croatian Higher Education Information System,

ISVU [23]. Together with some other data, taken from our course administration and statistics, they are summarized in the bottom rows of the Table 1.

The table shows the data for four academic years. The two grayed-out columns present the COVID-19 academic years: 2019/20 and 2020/21. To objectively compare them with the pre and post COVID-19 years, the academic years 2018/19 and 2021/22 are included. The latter is still current academic year in the moment of writing of this paper, its column is only partially filled.

The data included in the table is as follows: number of students attending the course in certain year, *N*; number of computer lab groups, N_{GR}; maximum number of attending students on lectures or labs in certain year – unofficial, internal teachers' data, N_{MAX}; number of teach hours (thr) for lectures on-site LEC_{SITE} and labs performed on-site LAB_{SITE}; number of teach hours (thr) for lecture and labs performed online, LEC_{ONLN} and LAB_{ONLN}. Total number of lecture and computer lab teach hours is calculated in row marked *TOT*. Last two rows include official data related to percentage of positive exam pass, *PASS*; and average grade AVG_{1_5} – in Croatia, negative grade is represented by 1, while highest grade, "excellent", is 5.

Number of teach hours is exact (with decimals since it was documented on minute basis), as performed. The standard teaching hour in Croatia lasts 45 minutes and that duration of teach hour was used for online lessons. However, contact lessons hold on-site in 2020/21 and 2021/22 use little "epidemiologically" redefined teach hours – in 2020/21 lasting for (only) 30 minutes, and in 2021/22 for 40 minutes.

Some additional explanations required to fully understand teach hours. When performing standard on-site labs (academic years 2018/19 and 2021/22), LAB_{SITE} corresponds to the total number of teach hours for all the groups together. In 2018/19, there were two groups (N_{GR}) defined for computer labs, while in 2021/22 three groups are defined. Also, please mark that 2021/22 labs are not finished yet, so that number of hours is not final (total of 45 teach hours is expected).

Table 1. Course related data – prior, during and post-COVID classes

Academic Year	***2018/19***	***2019/20***	***2020/21***	***2021/22****
N	41	40	45	51
N_{MAX}	32	26	32	34
N_{GR}	2	2	2	3
LEC_{SITE} *thr*	30.00	9.33	3.75	29.65
LAB_{SITE} *thr*	29.91	-	-	35.63
LEC_{ONLN} *thr*	-	20.56	24.49	-
LAB_{ONLN} *thr*	-	17.20	24.67	-
TOT *thr*	30.00 / 29.91	29.89 / 17.20	28.24 / 24.67	N/A
PASS	56%	69%	52%	N/A
AVG_{1_5}	3.24	3.61	3.31	N/A

* *still not completed, missing some data*

On the other hand, but quite related, when performing labs online, in 2019/20 and 2020/21, there can be seen that less hours (17.20 versus 24.67) were performed in 2019/20, even though number of groups was the same. That is why, due to our inexperience, in 2019/20, computer labs were performed synchronously for all students at the same time, without dividing groups – in that academic year, student were little more involved in writing homework assignments, so that additional (not included in the table) time was spent to check and correct these student materials.

Finally, to comment the students' overall success, it is obvious that student achievements in COVID-19 years are fully comparable to other years. The pass rate and the average grades are even slightly better in 2019/20. It is hard to precisely explain this little increase in student scores – it may be due their increased involvement in the learning process or any other reason. The scores that will be obtained in 2021/22 shall give us an additional sight in that matter.

5 CONCLUSION

After describing and analyzing our model of online teaching and e-learning in the introductory web-programming course during the two COVID-19 academic years, 2019/20 and 2020/21, we can summarize our subjective and objective findings. We have managed to teach all the course topics as in the previous years and attain the same volume and quality of the learning outcomes. Most of our students successfully managed to keep the track of all the course topics and successfully finalized their programming assignments.

The authors' first and subjective opinions were that this skill-based course was successfully taught using the presented e-learning model. It even seemed that the students participating the course were more focused on the given assignments and their programming projects during this crisis than before it! It remains unclear what is the true cause of this. A logical explanation is that our web programming is an elective course, for which it would be normal that the interest of the participating students is greater than in the mandatory courses.

Our extra efforts involved in preparing the technology for better quality video recordings of our synchronous online teaching — that were published also on YouTube — improved our students' e-learning experience and enlarged their motivation to learn the course. In the same time, we have benefited from being acquainted with the contemporary online teaching technology and from learning things that we would probably not have learned if there were no for the extreme pandemic situation of COVID-19.

Our subjective opinion that the student success was not different from the previous years was approved by their pass rate and the achieved average grades. This confirms that our approach in the online teaching that was presented in the paper proved successful even for the skill-based course as ours.

Of course, there are things that may be improved, but we are generally glad that the educational results were not diminished and that everything went well. Finally, we hope that our experiences will be useful to other educators and lecturers who prepare online teaching of the similar (programming) courses, but not because of some crisis like the one we just got out of.

REFERENCES

[1] M. Mikac, "Introductory mobile application development course for undergraduate students experiences", in *ICERI2020 Proceedings*, IATED Academy, pp. 5003-5011, 2020.

[2] M. Mikac, "Remote learning experiences with introductory mobile application development course", in *EDULEARN2021 Proceedings,* IATED Academy, pp. 2948-2953, 2021.

[3] M. Mikac, "An approach for creating asynchronous e-learning lectures by using materials recorded while performing synchronous learning", in *ICERI2020 Proceedings*, IATED Academy, pp. 4684-4693, 2020.

[4] *Visual Studio Code*. Accessed 05. May 2022. Retrieved from https://code.visualstudio.com

[5] XAMPP. Accessed 05. May 2022. Retrieved from https://www.apachefriends.org

[6] WAMP. Accessed 05. May 2022. Retrieved from https://www.wampserver.com/en/

[7] R. Logožar, M. Horvatić, I. Šumiga, M. Mikac, "Challenges in Teaching Assembly Language Programming — Desired Prerequisites vs. Students' Initial Knowledge", in *IEEE Global Engineering Education Conference (EDUCON2022) Proceedings*, IEEE, pp. 1688-1697, 2022.

[8] *Bootstrap.* Accessed 05. May 2022. Retrieved from https://getbootstrap.com

[9] *W3.CSS.* Accessed 05. May 2022. Retrieved from https://www.w3schools.com/w3css

[10] *Apache HTTP Server.* Accessed 05. May 2022. Retrieved from https://httpd.apache.org

[11] *MariaDB Database Server.* Accessed 05. May 2022. Retrieved from https://mariadb.org

[12] *MySQL Database Server*. Accessed 05. May 2022. Retrieved from https://www.mysql.com

[13] *phpMyAdmin*. Accessed 05. May 2022. Retrieved from https://www.phpmyadmin.net

[14] *MySQL Workbench*. Accessed 05. May 2022. Retrieved from https://www.mysql.com/products/workbench

[15] *Google Meet.* Accessed 05. May 2022. Retrieved from https://meet.google.com

[16] M. Mikac, "*matija.mikac-Online Education*" YouTube channel -YouTube course recordings. Accessed 05. May 2022. Retrieved from https://www.youtube.com/playlist?list=PLAdA7kbYHQ1jAv-tFCyuyXichbeCTwVFE

[17] *Merlin – Virtual e-learning environment.* Accessed 05. May 2022. Retrieved from https://moodle.srce.hr

[18] *Moodle – Learning Management System*, Accessed 05. May 2022. Retrieved from https://moodle.org/

[19] *OBS Studio – Open Broadcaster Software*. Accessed 06. May 2022. Retrieved from https://obsproject.com

[20] *HandBrake – Open Source Video Transcoder*. Accessed 06. May 2022. Retrieved from https://handbrake.fr

[21] *ShotCut Video Editor.* Accessed 06. May 2022. Retrieved from https://shotcut.org

[22] *Chrome Remote Desktop*. Accessed 06. May 2022. Retrieved from https://remotedesktop.google.com

[23] *ISVU* – Croatian High-Education Information System. Accessed 06. May 2022. Retrieved from https://www.isvu.hr