






CodeLab: An Online Laboratory for Learning to Code

Carles Garcia-Lopez^(✉) , Enric Mor , and Susanna Tesconi 

Universitat Oberta de Catalunya (UOC), 08018 Barcelona, Spain
{carlesgl, emor, stesconi}@uoc.edu

Abstract. Educational laboratories are flexible environments that allow learners to learn by practice, fostering their creativity, learning awareness, and collaboration with peers. Bringing these laboratory environments to an online setting is both challenging and necessary, particularly nowadays, when a significant part of learning takes place in online settings. Educational laboratories are well-suited places for learning to code, which is stated to require a great effort from learners, especially for non-STEM learners. This paper presents the design, development, and evaluation of CodeLab, a laboratory-based platform for learning to code through practice. A user-centered design approach was carried out, making learners active members of the design process through different design methods. As a result of two design iterations, CodeLab provides an integrated practice environment with a learning path based on a list of challenges and activities. Learners solve these activities and engage with their learning process by being aware of their own progress. The tool conveys a laboratory experience to non-STEM learners, fostering their practice skills, assessment, and autonomy.

Keywords: Design · Interaction design · User-centered design · Learning labs · Learning tools · Technology-enhanced learning · Learning to code

1 Introduction

Educational laboratories are flexible and multidisciplinary spaces where knowledge is built through social interaction and experimentation. In these spaces, students' creativity, self-training, and self-management are fostered, and the learning process is expanded to the educational community, developing and sharing learner's creativity with peers. Thus, a laboratory is an ideal place for testing and experimentation to learn from mistakes and acquire learning by practicing skills and learning by doing [1].

Currently, a relevant part of learning takes place in online settings. Therefore, there is a need to provide the learning community with an online laboratory that brings face-to-face laboratory's strengths to an online learning environment. However, although online learning has multiple advantages, it can sometimes lead to a disconnection between learners and teachers, promoting isolated and individual work. Designing and developing an online laboratory is a significant challenge that requires a deep understanding of users' needs and making them active participants in the design process to

create an environment that provides them with a good user experience and satisfaction to foster learners' engagement.

Learning to code is known to require a lot of practice [2, 3] since learners need to acquire coding and computational thinking skills and not just knowing the syntax of a particular programming language [3, 4]. In response to the requirements of the learning by doing approach [1], students need to monitor their learning process to direct, correct, and think deeply about their coding errors and solutions [6]. In such a challenging learning process, social interaction is also an essential component that allows students to share their knowledge, experiences, and questions with their peers and teachers while they are coding.

Given this major challenge of learning to code, continuous practice appears to be a viable approach to facilitate the learning process. In this respect, a laboratory-based tool becomes an essential tool to support the learning process in online settings.

CodeLab is a project that aims to create an online laboratory, offering students a workspace to practice, promoting the interaction between peers that occurs in a face-to-face laboratory, and fostering the awareness of their learning progress [6]. To do so, these key features were integrated into the same interface, providing students with a laboratory experience where they can easily access (1) learning content and activities, (2) progress and assessment feedback, (3) coding console, (4) execution and visualization and (5) a place to interact with other learners and teachers. From the instructors' perspective, CodeLab is expected to offer teachers an environment to keep track of students' activity and learning process and to provide feedback during the course.

CodeLab project follows a human-centered design (HCD) approach that aims to involve users in all phases of the design process [7] through an iterative process. This design approach is especially useful when designing interactive technologies [8] and technology-enhanced learning (TEL). Involving users in each step of the process, rather than waiting until the product is completed [9], helps to align the final product with users' needs and characteristics, which leads to a good learning experience [10].

This paper is organized as follows; the state of the art of learning to code by practice is presented in Sect. 2. Section 3 presents the approach to design and develop a learning tool to practice. The implementation and the evaluation of this tool are presented in Sect. 4. Finally, in Sect. 5, the conclusions and future work are discussed.

2 The State of the Art

As mentioned above, educational laboratories provide learners with a flexible environment for learning through practice, fostering creativity, learning awareness and interaction with peers and teachers. Recreating this favorable situation in an online setting can be very challenging. Commonly used learning management systems (LMS) provide a set of tools and features to scaffold the learning process, but in general they are mainly oriented to the acquisition of content than to the promotion of practice-based learning [11]. Technology-enhanced learning (TEL) research is trying to address

the need for more active and engaging online learning environments by proposing several tools and platforms for autonomous learning through practice [12–14]. Such tools are key elements for the design and implementation of online learning laboratories.

Literature is abundant on how challenging it can be to learn to code and the great effort students have to make during the learning process [15–18]. It can even be more challenging when those who have to learn to code are non-STEM students. Schachman [19] describes these students as “alternative programmers” since they do not identify themselves as programmers, but they need to program to achieve some of their goals.

Programming is widely understood as a creative and collaborative process between programmers and machines [19]. Thus, learning to code is not only about acquiring specific knowledge of the syntax but also about acquiring specific skills [20]. Among these skills, problem-solving is one of the most relevant ones [21], making students understand the context, identify key information, and plan to solve the problem [22, 23], emphasizing the ability to solve it by cooperating with their teachers and peers [24, 25]. Furthermore, time management is also a crucial skill that allows students to plan the use of time and the stages during the learning process [24, 26]. To acquire these programming and general skills, programming courses are generally characterized by providing students with many activities to practice coding intensively [27]. Students’ autonomy is crucial during this learning process since students need to learn to decide what is essential to succeed in their learning goals [28, 29].

Considering this need to practice intensively and how challenging it can be to learn to code to non-STEM students, it makes sense that these students might sometimes feel overwhelmed. In this context, student engagement takes on a significant relevance in their learning process. Engagement is understood as how actively students are involved in activities [30, 31], and there is stated to be a close connection between students’ engagement and the interactions between learners, teachers, and the learning environment [32, 33]. The learning platform plays a key role in an online course since it becomes the place where these interactions occur [34].

Given the importance of the learning platform in an online course, it makes sense to focus efforts on designing a useful and easy-to-use platform. In this regard, Davis remarks in [35] the connection between the level of perceived usefulness and the user behavior and intention to use it. In other words, the easier it is to use a platform, the less effort it will require from the users, making them more likely to accept and use it [35–37].

A human-centered design approach facilitates users to have a satisfactory first experience with the program and makes them willing to keep using it [9]. Interface and interaction design may also facilitate users to feel comfortable with the platform and its interface [38]. Interface takes on particular relevance on a platform to practice to code since there is a direct link between how programmers work and the design of the platform’s interface [19]. In this regard, discrepancies between how users and software designers understand the platform are inevitable [19]. However, being some of these discrepancies seen as the root of usability issues [39], it is important to design a satisfactory user experience, focusing on the interface design and its usability.

According to what has been exposed above, several design goals (DG) are identified to provide non-STEM students with a learning tool to practice coding that allows them:

- **DG1.** to practice autonomously solving provided activities,
- **DG2.** to split the activities into pieces,
- **DG3.** to be aware of their learning process,
- **DG4.** to communicate with their teachers and peers.

Next section details how these design goals were addressed through later stages of a user-centered design process, involving different types of users in the process through different design methods.

3 CodeLab Tool

CodeLab was conceived as a learning tool to promote learning through practice in an online laboratory setting. The tool provides non-STEM students with a platform that allows them to practice coding autonomously and collaboratively with their teachers and peers. The tool allows teachers to follow each student's particular learning progress and provide support. The interface was designed to foster student's exploration and discovery. According to Schachman [19], alternative programmers drive their work by feelings, intuitions, or emotions. Thus, CodeLab provides students with a list of activities for each challenge, where they can find assessment, recommended and complementary activities. These activities are shown with no visual-hierarchy difference.

3.1 Design Process

CodeLab was designed, developed, and evaluated through a user-centered design (UCD) approach, following the principles and phases of the ISO 9241-210 human-centered design process [40]. This is an iterative process divided into four main phases: (1) understand and specify the context of use; (2) specify the user requirements in sufficient detail to drive the design; (3) produce design solutions which meet these requirements and (4) conduct user-centered evaluations of these design solutions and modify the design taking into account the results. As Fig. 1 shows, different design methods and tools were used in each phase of the project during the first two iterations presented in this work.

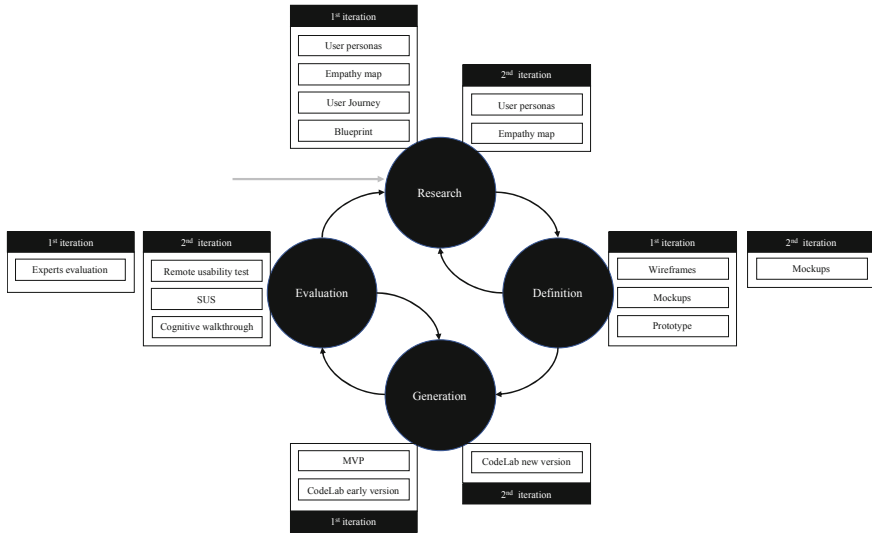


Fig. 1. Design process and methods.

As introduced above, the first phase of the design process focused on understanding the context and gathering information about the users. In this case, the final users are divided into two groups: non-stem students and teachers. This exploratory phase also aims to generate ideas, searching for diverse concepts and alternatives [41]. With this purpose, a collaborative workshop was carried out based on a design thinking process. As a starting point, two user personas were created to emphasize with the final user [42] and to understand these users' needs and goals that had to guide the design process [43–45]. During the two iterations of the UCD process, different methods that were used are described below:

Empathy Map. The empathy map (EM) method is an essential tool of a user-centered design approach [46]. Through an EM, people involved in the design process can understand, emphasize, and internalize a specific person's experience while using the product or service [47]. In this case, the updated version of Gray's initial EM was used, which consists of six areas: (1) See, (2) Say and Do, (3) Think and Feel, (4) Hear, (5) Pain, and (6) Gain [48].

It should be highlighted that most of the pains identified were related to the feeling of loneliness that a student might feel while learning to code in an online learning setting. Furthermore, they might feel that they have to make a great effort to achieve the goals and learn to code. On the other hand, the gains were related to their satisfaction with the results and the desire to share them with others.

User Journey. A User Journey (UJ) is a useful tool when a system is being developed from scratch [49]. A UJ aims to show, step by step, the interaction that users do while using the service or the product, describing emotions and reactions in each touchpoint. UJ usually considers the interaction that occurs before, during, and after using the service [50]. In this project, two UJ were built, focused on students and teachers (Fig. 2).

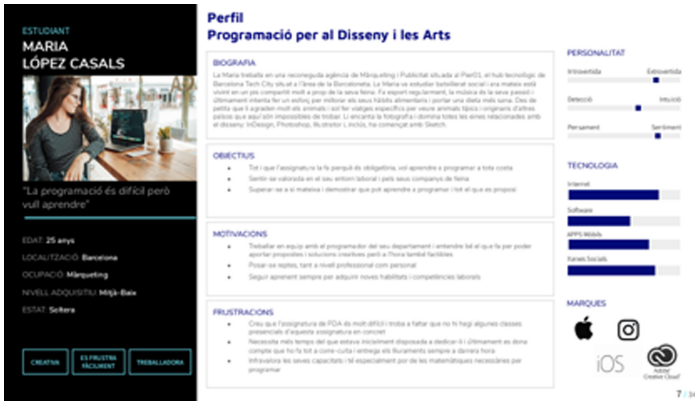


Fig. 3. The second version of the User Persona.

During the first iteration of the design process, the need to provide a broader view of the student experience was identified, from the information gathering phase to the completion of the course. In this regard, a new UJ was built, taking into account the five phases that a UOC student follows. In each phase, different touchpoints were identified (Table 1).

Table 1. Phases and touchpoints of the second UJ.

Phase	Touchpoints
Course registration	Course information, recommendations
Onboarding	Welcome, learning plan, and study guide
Methodology	Syllabus, learning resources, tools and programming environments
Support	Communication, mentoring
Evaluation	Evaluation criteria, feedback

In addition to the specific design opportunities identified, through the two iterations described above, it was also possible to identify a set of findings (F) about the students' main concerns in the whole process that they carry out during the course. These findings are listed below:

- **F1.** Students think the course contents and skills to be acquired are too complicated and sometimes do not know how to get started
- **F2.** Students miss being able to practice in the company of their teachers and peers
- **F3.** Students perceive an excessive workload, and it is often difficult to organize themselves
- **F4.** Students are overwhelmed by the frequency of activities submissions and find it difficult to organize their work
- **F5.** Students perceive an excessive workload, and it is often difficult to organize themselves

- **F6.** Students feel that it is difficult to communicate with their peers and the teacher due to the high number of students
- **F7.** Students believe that the forum of the course is underutilized
- **F8.** Students feel they cannot make enough progress as they do not have direct feedback.

3.2 Conceptualization and Design

During the first phases of the design process exposed in Sect. 3.1 and the literature research presented in Sect. 2, different findings (F) and design goals (DG) were identified. In the generation phase of the project, design features (DF) were set, taking into account the findings and design goals (see Fig. 4).

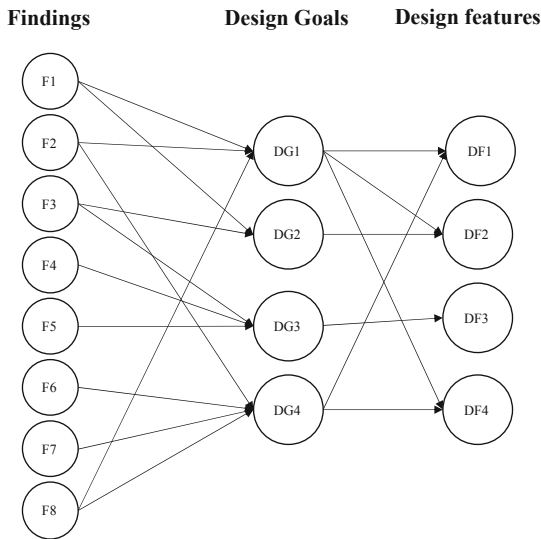


Fig. 4. Relation between findings (F), design goals (DG) and design features (DF).

These design features (DF) are detailed below:

DF1. Laboratory-Based Environment. Providing students with a platform to practice programming follows the idea of providing them with a programming laboratory. However, mirroring the dynamics that occur in a face-to-face laboratory to an online environment can be challenging. The CodeLab experience is designed to give learners a feeling of being in a face-to-face laboratory, where they can practice while interacting with their teachers and peers. Thus, the workspace organization and design are a crucial feature to be addressed. In this regard, CodeLab’s interface brings together diverse elements that a student might expect from a practice lab (see Fig. 7): (1) a contextual navigation with information about the exercise, (2) information about the progress, (3) an area to write code, (4) an area to visualize code execution, and (5) an area to share the experience with peers and teachers. Even so, it is important to allow learners

to adapt the interface to their needs at any given moment. Thus, CodeLab’s programming screen is designed to be adaptable to the students’ needs, being able to minimize some of the previously mentioned areas (see Fig. 5).

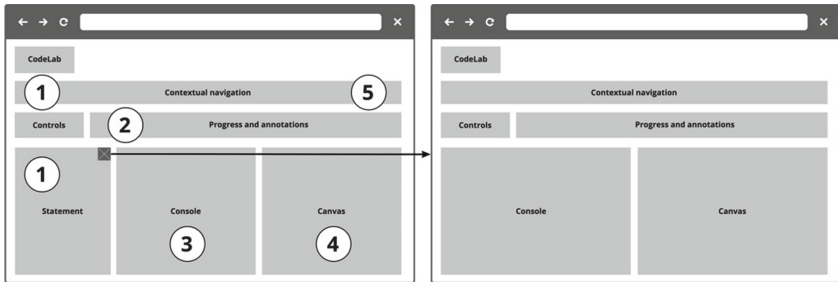


Fig. 5. Lo-fi wireframe of CodeLab’s programming screen.

DF2. Challenges and Activities. Learning to code is not just about doing specific activities. The acquisition of programming skills might transcend the fact of solving individual activities and be based on practicing a group of them. In this regard, CodeLab classifies activities into different challenges designed by teachers. These challenges ensure that students will get the knowledge when they finish them. Furthermore, the platform must be sufficiently flexible to allow students to guide their own learning process, allowing them to go into more or less depth on the topics they find necessary. In this sense, it was decided to provide all the activities in each challenge openly and not to block some of them depending on the evolution of the student, fostering their autonomy.

However, in order to facilitate a learning path to students, activities are classified according to two different criteria: difficulty and type of activity. First, teachers indicate difficulty in the three-point Likert scale, one of the easiest and three most difficult ones. Secondly, in order to balance the autonomy and the assessment of their studies, exercises are also tagged to whether they are “recommended”, “complimentary”, or “assessment” (see Fig. 6).

Activity	Difficulty	Type
1. Name of the activity - Name of the Challenge	● ○ ○	C
2. Name of the activity - Name of the Challenge	● ● ○	e
3. Name of the activity - Name of the Challenge	● ○ ○	C

Fig. 6. Lo-Fi wireframe of the classification of activities according to their difficulty and typology.

DF3. Learning Awareness. A platform focused on practicing should include design elements that allow students to be aware of their position on the learning path. These features were structured hierarchically, from each challenge and activities’ progress to the course’s general progress. Regarding each exercise’s progress, a progress bar was designed to be shown on the screen where they practice (Fig. 7). This progress bar shows the progress of the exercise divided into different steps that have been set by teachers, allowing students to mark their own progress and add personal notes in each step.

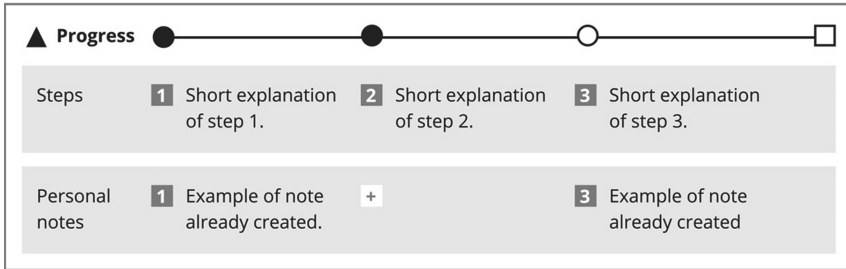


Fig. 7. Lo-Fi wireframes of the progress bar in the activity screen.

On top of that, students can see their general progress from the CodeLab homepage, where they can see at a glance what their overall progress is and access each challenge and activity directly (Fig. 8).

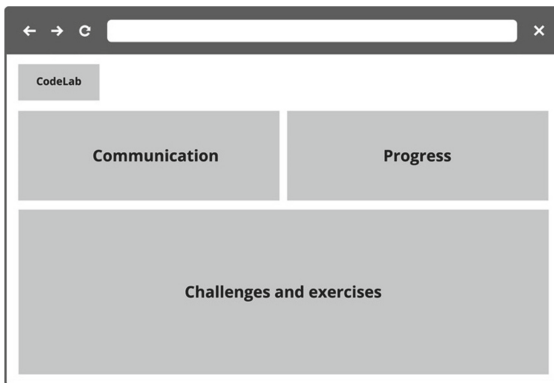


Fig. 8. Lo-Fi wireframes of CodeLab’s homepage.

DF4. Scaffolding and Collaboration. During a self-paced practice-based learning process, students are expected to carry out a significant number of editing and compilations before submitting the activity. At this point, it is important to mention the educational settings where CodeLab will be used. During the course, students must submit some assessment activities on specific dates, which are part of the continuous

assessment grade. Thus, between these submission dates, students are expected to practice autonomously. In some cases, students might feel that they do not receive feedback until the task is done and submitted, making it challenging to understand the problem [20]. In this regard, taking into account a commonly used feature in Integrated Development Environments (IDEs), it was decided to include feedback to the students when they compile the code, highlighting the line or lines where the error is.

4 Development and Evaluation

CodeLab was designed, developed, and implemented at Universitat Oberta de Catalunya (UOC), an entirely online higher education institution based in Barcelona (Spain).

4.1 Development

CodeLab was conceived as a laboratory-based tool potentially connectable to any learning management system (LMS). To do that, it was developed using the IMS LTI standard [55]. To achieve this laboratory-based environment, wireframes that were designed emphasized the idea of providing learners with a single space where they could find all the resources they would find in a face-to-face laboratory. Once the wireframes were evaluated by experts (interaction designers and programming teachers), they were evolved into mockups that embraced UOC's style guide look and feel (Fig. 9). From a technical point of view, the development of the interface is based on the VueJS framework, the back-end is based on Java SpringBoot, challenges and activities are stored on a MySQL database and learner progress and submitted activities are stored on a GitLab.

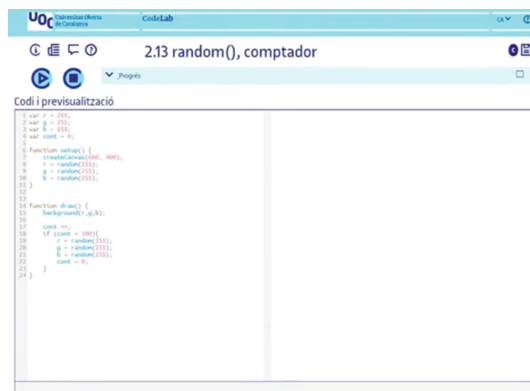


Fig. 9. CodeLab's screen to practice coding.

As explained in Sect. 3.2, one of the key features in this CodeLab version was to show activities classified by challenges to foster learners' exploration. In this sense, it

was decided to show the level of difficulty and the activity type. In addition, it was decided to add a progress bar in each activity to facilitate students to be aware of their learning progress. Figure 10 shows how these activities were shown after the visual implementation.



Fig. 10. List of activities in CodeLab.

Another key aspect of fostering learners’ autonomy is the progress bar shown in each activity. This bar is divided into two parts, the first one is where the learner can see each step of the activity and mark it as completed, and the second one is the space where personal annotations can be made (Fig. 11).

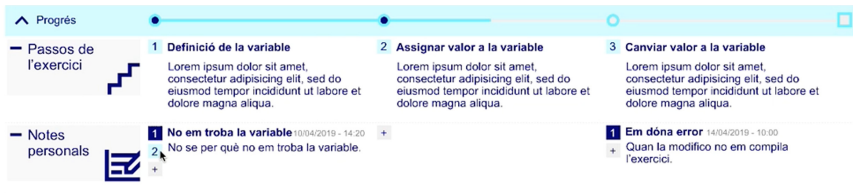


Fig. 11. Progress bar into each activity screen.

Due to the project scope, the communication functionality was postponed for the next development iteration, and students used the classical communication channels provided by the UOC’s virtual campus.

4.2 Evaluation

The evaluation of this functional version of the tool was based on different design methods with different participants to evaluate the design and development of the platform (see Table 2).

Table 2. Evaluation methods and participants.

Method	Type	Participants
Cognitive walkthrough	Remote test	Teachers and experts
System Usability Scale (SUS)	Remote questionnaire	Students
Usability test	Remote test	Students

Cognitive Walkthrough (CWT). A CWT is a design method to inspect the usability of a platform proven to be useful for identifying problems with navigation and information search into the platform [56]. Through this method, participants are asked to perform different tasks and answer four questions in each task [57]:

- Q1. Will the user try to achieve the right effect?
- Q2. Will the user notice that the correct action is available?
- Q3. Will the user associate the correct action with the effect that the user is trying to achieve?
- Q4. If the correct action is performed, will the user see that progress is being made toward the task's solution?

A total of 6 evaluators performed 10 tasks and sub-tasks on the CodeLab platform and answered these questions. There were three tasks with a rate of success lower than 50% (see Table 3). The feedback that participants provided in each of these tasks arose some usability issues that needed to be addressed:

- **Task 2. Enter a complementary activity of the “Challenge 2”.** Participants who failed in this task indicated that it was challenging to interpret the tag that indicates the type of exercise.
- **Task 2.1. Read the activity. Close the statement and open it again.** All the participants who failed this task agree that it was difficult to understand how to open the exercise again. Some of them suggested that the icon should be more precise.
- **Task 4. Do steps 1 and 2 of activity 1.21. When finished, mark the progress in the progress bar and save the activity.** Some of the participants who failed in this task explained that the steps were not visible enough and how to mark the progress should be improved.
- **Task 4.1. Save the activity. Close it and go back to CodeLab's homepage.** The main problem in this task was related to the closure of the exercises. All the participants expressed that they did not know how to close it. Furthermore, there was a problem when going back to the CodeLab's homepage because some did not know how to do it.

Table 3. Percentage of success of the tasks in the cognitive walkthrough

Task	Q1	Q2	Q3	Q4
1	100	72	75	100
2	69.23	46.15	63.64	77.78
2.1	75	33.33	63.64	90
3	91.67	71.43	81.82	58.33
4	75	40	66.67	72.73
4.1	41.67	50	50	50
5	84.62	91.67	91.67	91.67
5.1	83.33	83.33	83.33	91.67
5.2	75	83.33	75	91.67
5.3	91.67	75	75	81.82

System Usability Scale (SUS). The SUS questionnaire is widely used for evaluating the usability perceived by participants [58, 59], providing a general overview of the usability of the platform [60]. Participants are asked to answer 10 5 Likert-scale questions after having used the platform [61].

The result of SUS was calculated (1) and resulted in an average score of 84.69 (see Fig. 9), which is an acceptable score in terms of usability [62] (Fig. 12).

$$SUS = 2.5(20 + \text{SUM}(SUS01, SUS03, SUS05, SUS07, SUS09) - \text{SUM}(SUS02, SUS04, SUS06, SUS08, SUS10)) \quad (1)$$

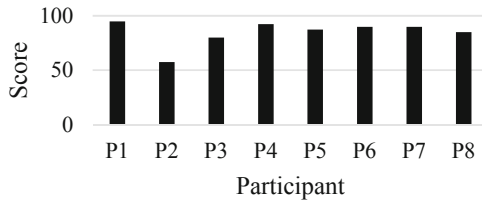


Fig. 12. Results of the system usability scale

However, since the SUS method was designed to be an additional method to complement objective methods [63], remote usability tests were also performed with users.

Remote Usability Tests. A total of 8 students were asked to perform 6 tasks and sub-tasks on the CodeLab platform. During this test, the moderator asked the participant to carry out predefined tasks on the CodeLab platform. This process was captured to be later analyzed quantitatively and qualitatively. Firstly, two of the metrics proposed by Harrati et al. in [59] were used: Completion rate and task duration. Secondly, the moderator took notes about the reactions and comments of the participants while performing the task. The tasks are listed below:

- **Task 1.** Enter the CodeLab tool with your user and password. Take a look at the first page. Open the recommended activity named “Activity 1” and go back to CodeLab’s homepage.
- **Task 2.** Enter the complementary activity named “Activity 2 – Automats”. How would you close (hide) the activity statement? Once it is closed, how would you open it again?
- **Task 3.** Solve activity 1.5b. Save it and be sure it has been appropriately saved.
- **Task 4.** Solve steps 1 and 2 of activity 1.21. When finished, mark the progress in the progress bar and save the activity. Go back to the home page.
- **Task 5.** Enter again activity 1.21 and add a comment.
- **Task 6.** Do the two last steps of exercise 1.21 and check that it works. Mark it as completed Do the two last steps of activity 1.21 and check that it works. Mark it as completed and save it.

Results showed that 50% of the tasks were finished by all the participants (see Table 4). Task1, Task 2 and Task 4 did not achieve a 100% completion rate. Task 1 was finished by 50% of participants, and most of the participants had problems when trying to go back to the CodeLab homepage, which made them fail the task. Furthermore, 37.5% reported difficulties understanding the type of the activity. Task 2 was completed by 75% of participants, 25% who failed did not know how to see the statement of the activity when closed. Lastly, Task 4 was completed by % of the participants, 37.5% of them did not pay attention to the exercises' steps and did not see them until the moderator indicated where to find them.

Regarding the average time they spent on doing the tasks, no issue was identified since those tasks that took more time were the ones where participants had to do a real activity.

Table 4. Performance of each task in the test with users.

Task	Completion rate (%)	Average task duration (mm:ss)
1	50	01:53
2	75	01:20
3	100	04:36
4	62.5	07:01
5	100	01:23
6	100	02:16

5 Conclusion and Future Work

This work presents the design process of the CodeLab tool, a laboratory-based platform that allows students to practice during a programming course at the Open University of Catalonia. A key aspect of CodeLab is to convey a similar experience to what students might get in a face-to-face laboratory. The learning tool was designed and evaluated with experts, teachers, and students through a user-centered design approach.

Although some functionalities identified in the early design phases were postponed to future iterations, a fully functional version of CodeLab was designed, developed and evaluated in two different courses.

The results obtained from the design process and the evaluation methods provided important information about the usability and user experience of the platform, pointing to specific design requirements to be addressed. It is worth noting that, although some areas of improvement were uncovered during the internal work process, carrying out evaluation methods through a UCD approach facilitated obtaining detailed information that otherwise would probably not have been possible to obtain.

First, the classification of each learning activity according to its difficulty and typology needs to be more explicit to the user. The results in Task 2 of the CWT arose a problem with the label used to classify each activity, and 37.5% of participants in the usability test pointed to the same issue. Thus, the interface design should organize learning activities in the general list of activities in a more visible way by revising the graphic elements of the interface.

Secondly, an area of improvement was also identified in the progress bar of each activity. Task 4 of the CWT, which involved the progress bar and steps of the learning activities, had the lowest success rate in the CWT. Participants also expressed that the progress bar and the steps were not clear enough on the activity interface. In this sense, 37.5% of participants failed Task 4 on the usability test, where they could not locate activity's steps. Hence, the progress bar must be redesigned in the next iteration of the project.

Finally, in addition to improvements to the functionalities already implemented, it is necessary to address the integration of a direct communication tool through the platform itself to move towards a laboratory-based experience.

In conclusion, the importance of providing tools that facilitate practicing to learners has been highlighted in this work, emphasizing the relevance in a context with non-STEM students who need to learn to code. Designing, developing, and evaluating this laboratory-based platform was a significant challenge that has been addressed and achieved through a user-centered design approach, leading to a practice-based tool that adapts to the learners' needs, resulting in a good user experience and satisfaction.

Since the design goals presented in this work have been successfully addressed, future work should further develop the idea of an online learning laboratory-based setting: developing analytics features that allow teachers to better support learners, implementing a communication feature in CodeLab to foster collaborative practice between peers and teachers, and improving the interface design of the platform. Since the two first iterations have been centered on evaluating the platform's development, the following iterations must focus on the evaluation of the implementation to understand the impact it might have on learners' engagement and learning outcomes.

References

1. Romero, M., Lepage, A., Lille, B.: Computational thinking development through creative programming in higher education. *Int. J. Educ. Technol. High. Educ.* **14**(1), 1–15 (2017). <https://doi.org/10.1186/s41239-017-0080-z>
2. Gomes, A., Mendes, A.J.: Learning to program - difficulties and solutions | Academic Conference Paper, no. May, 2007. https://www.researchgate.net/publication/228328491_Learning_to_program_-_difficulties_and_solutions
3. Lahtinen, E., Ala-Mutka, K., Järvinen, H.-M.: A study of the difficulties of novice programmers. *SIGCSE Bull.* **37**(3), 14–18 (2005). <https://doi.org/10.1145/1151954.1067453>
4. García-Peñalvo, F.J., Mendes, A.J.: Exploring the computational thinking effects in pre-university education. *Comput. Hum. Behav.* **80**, 407–411 (2018). <https://doi.org/10.1016/j.chb.2017.12.005>
5. Havenga, M., et al.: Metacognitive and problem-solving skills to promote self-directed learning in computer programming : teachers ' experiences. *SA-eDUC J.* **10**(2), 1–14 (2013)
6. Mor, E., Santanach, F., Tesconi, S., Casado, C.: CodeLab: designing a conversation-based educational tool for learning to code. In: Stephanidis, C. (ed.) *HCI 2018. CCIS*, vol. 852, pp. 94–101. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-92285-0_14
7. Norman, D.A., Anderson, N.S., Norman, D.A., Draper, S.W.: User centered system design : new perspectives on human-computer interaction to cite this version : HAL Id : hal-00190545. *Am. J. Psychol.* **101**(1), 148 (1986)

8. Lowdermilk, T., Design, U.-C.: A Developer's Guide to Building User-Friendly Applications. O'Reilly Media, Sebastopol (2013)
9. Leonard, B., Vincenti, G.: Engaging programming students through simpler user interfaces. In: Karwowski, W., Ahram, T., Nazir, S. (eds.) AHFE 2019. AISC, vol. 963, pp. 113–121. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-20135-7_11
10. Garreta Domingo, M., Mor Pera, E.: User Centered Design in E-Learning Environments: from Usability to Learner Experience (2007)
11. Kasim, N.N.M., Khalid, F.: Choosing the right learning management system (LMS) for the higher education institution context: a systematic review. *Int. J. Emerg. Technol. Learn. (iJET)* **11**(06), 55–61 (2016). <https://online-journals.org/index.php/i-jet/article/view/5644>.
12. Duval, E., Sharples, M., Sutherland, R.: Research themes in technology enhanced learning. In: Duval, E., Sharples, M., Sutherland, R. (eds.) *Technology Enhanced Learning*, pp. 1–10. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-02600-8_1
13. Goodyear, P., Retalis, S.: *Learning, technology and design*. Leiden, The Netherlands: Brill | Sense, pp. 1–27
14. Cook, D.A., Ellaway, R.H.: Evaluating technology-enhanced learning: a comprehensive framework. *Med. Teach.* **37**(10), 961–970 (2015). <https://doi.org/10.3109/0142159X.2015.1009024>
15. Medeiros, R.P., Ramalho, G.L., Falcão, T.P.: A systematic literature review on teaching and learning introductory programming in higher education. *IEEE Trans. Educ.* **62**(2), 77–90 (2019). <https://doi.org/10.1109/TE.2018.2864133>
16. Robins, A., Rountree, J., Rountree, N.: Learning and teaching programming: a review and discussion. *Comput. Sci. Educ.* **13**(2), 137–172 (2003). <https://doi.org/10.1076/cs.ed.13.2.137.14200>
17. Pears, A., et al.: A survey of literature on the teaching of introductory programming. *SIGCSE Bull.* **39**(4), 204–223 (2007). <https://doi.org/10.1145/1345375.1345441>
18. Vihavainen, A., Airaksinen, J., Watson, C.: a systematic review of approaches for teaching introductory programming and their influence on success. In: *Proceedings of the Tenth Annual Conference on International Computing Education Research*, pp. 19–26 (2014). <https://doi.org/10.1145/2632320.2632349>
19. Schachman, T.: Alternative programming interfaces for alternative programmers. In: *SPLASH 2012: Onward! 2012 - Proceedings of the ACM International Symposium on New Ideas, New Paradigms, and Reflections on Programming and Software*, pp. 1–10 (2012). <https://doi.org/10.1145/2384592.2384594>
20. Lui, A.K., Kwan, R., Poon, M., Cheung, Y.H.Y.: Saving weak programming students: applying constructivism in a first programming course. *SIGCSE Bull. (Assoc. Comput. Mach. Spec. Interest Group Comput. Sci. Educ.)* **36**(2), 72–76 (2004). <https://doi.org/10.1145/1024338.1024376>
21. Medeiros, R.P., Ramalho, G.L., Falcao, T.P.: A systematic literature review on teaching and learning introductory programming in higher education. *IEEE Trans. Educ.* **62**(2), 77–90 (2019). <https://doi.org/10.1109/TE.2018.2864133>
22. Rusczyk, R.: *The art of Problem Solving, volume 1: The basics*. AoPS Incorporated, California (2006)
23. García-Peñalvo, F.J.: A brief introduction to TACCLE 3 — coding European project. In: *2016 International Symposium on Computers in Education (SIIE)*, pp. 1–4 (2016). <https://doi.org/10.1109/SIIE.2016.7751876>
24. de Lira Tavares, O., de Menezes, C.S., de Nevado, R.A.: Pedagogical architectures to support the process of teaching and learning of computer programming. In: *2012 Frontiers in Education Conference Proceedings*, pp. 1–6, October 2012. <https://doi.org/10.1109/FIE.2012.6462427>

25. Fuller, M.T.: ISTE standards for students, digital learners, and online Learning. In: *Handbook of Research on Digital Learning*, IGI Global, pp. 284–290 (2020)
26. Bati, T.B., Gelderblom, H., van Biljon, J.: A blended learning approach for teaching computer programming: design for large classes in Sub-Saharan Africa. *Comput. Sci. Educ.* **24**(1), 71–99 (2014). <https://doi.org/10.1080/08993408.2014.897850>
27. Lam, M.S.W., Chan, E.Y.K., Lee, V.C.S., Yu, Y.T.: designing an automatic debugging assistant for improving the learning of computer programming. In: *Hybrid Learning and Education*, pp. 359–370 (2008)
28. Hannafin, M.J., Hill, J.R., Land, S.M., Lee, E.: Student-centered, open learning environments: research, theory, and practice. In: Spector, J.M., Merrill, M.D., Elen, J., Bishop, M. J. (eds.) *Handbook of Research on Educational Communications and Technology*, pp. 641–651. Springer, New York (2014)
29. Glasgow, N.A.: *New curriculum for new times: a guide to student-centered, problem-based learning*. ERIC (1997)
30. Appleton, J.J., Christenson, S.L., Kim, D., Reschly, A.L.: Measuring cognitive and psychological engagement: validation of the Student Engagement Instrument. *J. Sch. Psychol.* **44**(5), 427–445 (2006). <https://doi.org/10.1016/j.jsp.2006.04.002>
31. Subramanian, K., Budhrani, K.: Influence of course design on student engagement and motivation in an online course. In: *Proceedings of the 51st ACM Technical Symposium on Computer Science Education*, pp. 303–308 (2020). <https://doi.org/10.1145/3328778.3366828>
32. Lear, J., Anson, C., Steckelberg, A.: Interactivity/community process model for the online education environment. ... *Online Learn. Teach.* **6**(1), 71–77 (2010). http://jolt.merlot.org/vol6no1/lear_0310.htm
33. Handelsman, M.M., Briggs, W.L., Sullivan, N., Towler, A.: A measure of college student course engagement. *J. Educ. Res.* **98**(3), 184–192 (2005). <https://doi.org/10.3200/JOER.98.3.184-192>
34. Reese, S.A.: Online learning environments in higher education: connectivism vs. dissociation. *Educ. Inf. Technol.* **20**(3), 579–588 (2014). <https://doi.org/10.1007/s10639-013-9303-7>
35. Davis, F.D.: Perceived usefulness, perceived ease of use, and user acceptance of information technology. *MIS Q.* **13**(3), 319–340 (1989). <http://www.jstor.org/stable/249008>
36. Lee, M.K.O., Cheung, C.M.K., Chen, Z.: Acceptance of Internet-based learning medium: the role of extrinsic and intrinsic motivation. *Inf. Manage.* **42**(8), 1095–1104 (2005). <https://doi.org/10.1016/j.im.2003.10.007>
37. Chen, W.S., Tat Yao, A.Y.: An empirical evaluation of critical factors influencing learner satisfaction in blended learning: a pilot study. *Univ. J. Educ. Res.* **4**(7), 1667–1671 (2016). <https://doi.org/10.13189/ujer.2016.040719>
38. Mor, Y., Winters, N.: Design approaches in technology-enhanced learning. *Interact. Learn. Environ.* **15**(1), 61–75 (2007). <https://doi.org/10.1080/10494820601044236>
39. Norman, D.: *The Design of Everyday Things* (2016)
40. I. O. for Standardization, *Ergonomics of Human-system Interaction: Part 210: Human-centred Design for Interactive Systems*. ISO (2010)
41. Perttula, M.K., Liikkanen, L.A.: Structural tendencies and exposure effects in design idea generation. In: *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, vol. 42584, pp. 199–210 (2006)
42. Acuña, S.T., Castro, J.W., Juristo, N.: A HCI technique for improving requirements elicitation. *Inf. Softw. Technol.* **54**(12), 1357–1375 (2012). <https://doi.org/10.1016/j.infsof.2012.07.011>
43. Adlin, T., Pruitt, J.: *The Essential Persona Lifecycle: Your Guide to Building and Using Personas* (2010)

44. Castro, J.W., Acuña, S.T., Juristo, N.: Enriching requirements analysis with the personas technique. In: CEUR Workshop Proceedings, vol. 407, no. June 2014 (2008)
45. Miaskiewicz, T., Kozar, K.A.: Personas and user-centered design: how can personas benefit product design processes? *Des. Stud.* **32**(5), 417–430 (2011). <https://doi.org/10.1016/j.destud.2011.03.003>
46. Bratsberg, H.M.: Empathy maps of the foursight preferences. In: International Center for Studies in Creativity (2012)
47. Gray, D., Brown, S., Macanuso, J.: *Gamestorming: A Playbook for Innovators, Rulebreakers, and Changemakers*. O'Reilly Media, Inc., Sebastopol (2010)
48. Gray, D.: Empathy map. In: Osterwalder, A., Pigneur, Y. (eds.) *Business Model Generation: A Handbook for Visionaries, Game Changers and Challengers*. Wiley, Hoboken (2010)
49. Caddick, R., Cable, S.: *Communicating the User Experience: A Practical Guide for Creating Useful UX Documentation*. Wiley, Hoboken (2011)
50. Hanington, B., Martin, B.: *Universal Methods of Design: 100 Ways to Research Complex Problems, Develop Innovative Ideas, and Design Effective Solutions*. Rockport Publishers, Beverly (2012)
51. Patrício, L., Fisk, R.P., Cunha, J.F.: Designing multi-interface service experiences: the service experience blueprint. *J. Serv. Res.* **10**(4), 318–334 (2008). <https://doi.org/10.1177/1094670508314264>
52. Münch, J., Fagerholm, F., Johnson, P., Pirttilahti, J., Torkkel, J., Jäärinen, J.: Creating minimum viable products in industry-academia collaborations. In: *Lean Enterprise Software and Systems*, pp. 137–151 (2013)
53. Edison, H., Wang, X., Abrahamsson, P.: *Lean Startup: Why Large Software Companies Should Care* (2015). <https://doi.org/10.1145/2764979.2764981>
54. York, J.L., Danes, J.E.: Customer development, innovation, and decision-making biases in the lean startup. *J. Small Bus. Strategy* **24**(2), 21–40 (2014). <https://libjournals.mtsu.edu/index.php/jsbs/article/view/191>
55. I. M. S. G. L. Consortium and others: *Learning Tools Interoperability Core Specification. IMS Final Release Version, vol. 1* (2019)
56. Blackmon, M.H., Kitajima, M., Polson, P.G.: Repairing usability problems identified by the cognitive walkthrough for the web (5), 497 (2003). <https://doi.org/10.1145/642696.642698>
57. Blackmon, M.H., Polson, P.G., Kitajima, M., Lewis, C.: Cognitive walkthrough for the Web. In: *Conference on Human Factors in Computing Systems - Proceedings*, vol. 4, no. 1, pp. 463–470 (2002). <https://doi.org/10.1145/503457.503459>
58. Lewis, J.R.: The system usability scale: past, present, and future. *Int. J. Hum. Comput. Interac.* **34**(7), 577–590 (2018). <https://doi.org/10.1080/10447318.2018.1455307>
59. Harrati, N., Bouchrika, I., Tari, A., Ladjailia, A.: Exploring user satisfaction for e-learning systems via usage-based metrics and system usability scale analysis. *Comput. Hum. Behav.* **61**, 463–471 (2016). <https://doi.org/10.1016/j.chb.2016.03.051>
60. Kaya, A., Ozturk, R., Altin Gumussoy, C.: Usability measurement of mobile applications with system usability scale (SUS). In: *Industrial Engineering in the Big Data Era*, pp. 389–400 (2019)
61. Brooke, J.: SUS: a “quick and dirty” usability. *Usability evaluation in industry*, pp. 189–194 (1996)
62. Bangor, A., Staff, T., Kortum, P., Miller, J., Staff, T.: Determining what individual SUS scores mean: adding an adjective rating scale. *J. Usability Stud.* **4**(3), 114–123 (2009)
63. Brooke, J.: SUS: a retrospective. *J. Usability Stud.* **8**(2), 29–40 (2013)