# REIT-Builder: Customizable Training for Requirements Elicitation Interviews

Roger Ian Konlog, Paola Spoletini

*Kennesaw State University, Marietta, GA, USA*

Abstract
Experiential learning and hands-on activities have been proven effective tools for teaching software engineering activities as they require a variety of different expertise not always effectively taught through traditional lectures. Among these activities are requirements elicitation interviews, for which studies have shown that role-playing and self- and peer-assessments can be successfully used to train students and young analysts. However, while effective, the training approaches proposed in the literature are "static" and assume that the instructor using them can allocate a fixed amount of time and human resources. On the other hand, developing personalized training is a time-consuming activity that might not result in effective programs. To overcome these limitations, we propose REIT-Builder (Requirements Elicitation Interviews Training Builder), a tool to support the development of effective training programs that are compatible with the instructor's available resources. In this tool paper, we present REIT-Builder, its architecture, the flow of interactions with the users, and the planned evaluation.

**Keywords**
Requirements Elicitation Interviews, Requirements Engineering Education and Training

## 1. Introduction and Motivation

The use of active and experiential learning has been extensively studied and encouraged as a teaching practice for software engineering due to its positive impact on students' engagement, the development of critical thinking skills, and the student's ability to collaborate in projects [1, 2, 3, 4]. In particular, such practices can be beneficial in facilitating the learning of complex concepts and processes, which might not be difficult in theory but cannot be easily understood in traditional lecture settings. Active learning approaches, such as case studies, simulations, and role-playing, can help students to better understand these concepts and apply them in practice. This is the case in many practices in requirements engineering (RE) [5, 6] and especially requirements elicitation interviews.

Indeed, many of the factors influencing the success of interviews, such as analysts' communication skills, cannot be effectively taught through lectures but can be acquired through practice[7, 8]. The effectiveness of analysts in conducting interviews highly depends on their experience and active participation in real(istic) interviews [9]. As mistakes made during the

*In: A. Ferrari, B. Penzenstadler, I. Hadar, S. Oyedeji, S. Abualhaija, A. Vogelsang, G. Deshpande, A. Rachmann, J. Gulden, A. Wohlgemuth, A. Hess, S. Fricker, R. Guizzardi, J. Horkoff, A. Perini, A. Susi, O. Karras, A. Moreira, F. Dalpiaz, P. Spoletini, D. Amyot. Joint Proceedings of REFSQ-2023 Workshops, Doctoral Symposium, Posters & Tools Track, and Journal Early Feedback Track. Co-located with REFSQ 2023. Barcelona, Catalunya, Spain, April 17, 2023.*

rkonlog@students.kennesaw.edu (R. I. Konlog); pspoleti@kennesaw.edu (P. Spoletini)

design and execution of interview tasks can have an impact on the resulting software and system requirements [10], learning how to perform an effective interview should be one of the primary objectives of RE courses. Literature offers a set of best practices to use active learning activities [11] and some more complex training programs to learn how to conduct an interview [12]. However, existing training might not be suitable for all specific settings given the potential differences in available resources (e.g., in terms of time or supporting personnel). Thus, instructors might need to create their own training by either adapting existing ones to their resources and needs [13] or building new ones from scratch. This could be time-consuming as many different activities and customization of such activities are possible.

To support instructors in developing personalized training for requirements elicitation interviews suitable for their resources, we propose REIT-Builder (Requirements Elicitation Interviews Training-Builder), a web application designed to provide support to instructors and trainers while building a training suitable for their course and their educational goals. REIT-Builder is an interactive system that allows users to create a training program to teach requirements elicitation interviews by choosing among a set of pre-defined activities (extracted by the literature and active and experiential learning best practices) or defining new ones. The different activities can be included in a program by using existing resources (made available by REIT-Builder) or be customized. While the user creates its training, REIT-Builder gives tips on the possible next activity to add, keeps track of the used resources in terms of students' and instructors' time to check if they are compatible with the user's actual resources, and alerts the users when needed. The initial version of REIT-Builder allows users to add, customize, and define activities and provides (a limited number of) tips to guide the users. It is available at https://reit-builder.web.app/.

## 2. Overview and Implementation Details

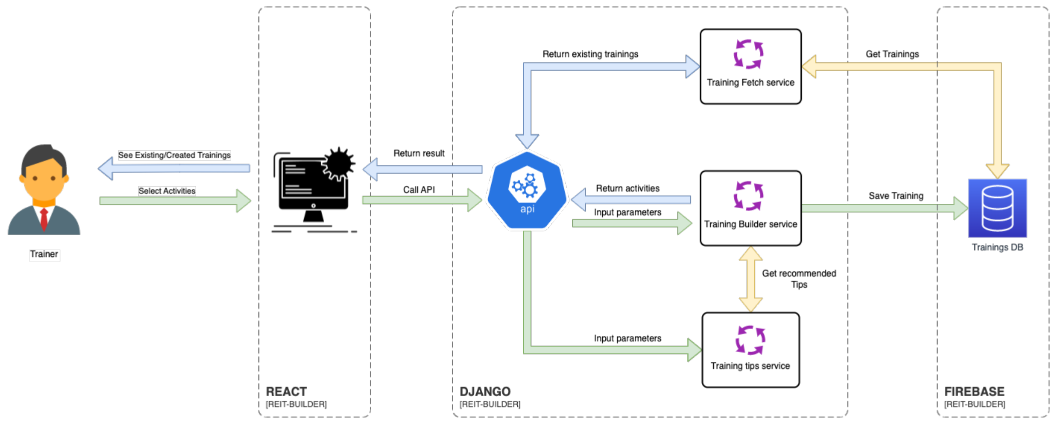


**Figure 1:** Overview of REIT-BUILDER.

REIT-Builder[1] is a web application that supports users in customizing training programs or reusing programs developed by other users (or already stored in the tool, as for example [12]).

[1]The project is available at https://github.com/IanKonlog/REIT

To enable these two functionalities, REIT-BUILDER provides a framework for (1) receiving input parameters from the users, (2) allowing the selection of the activities to be added to the program under development, (3) customizing and modifying activities, (4) providing tips for the selections of the following activities and (5) displaying the training programs to the user.

To implement REIT-BUILDER, we leveraged three popular technologies: React JS [14], a JavaScript library for creating user interfaces, Django [15], a Python web framework for developing back-end applications, and Firebase [16], a cloud-based platform that provides back-end services.

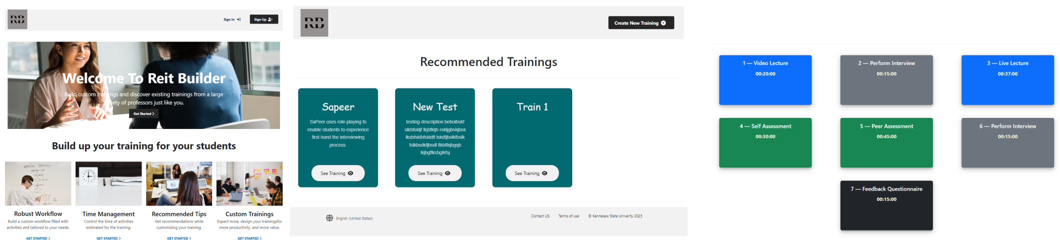
To perform back-end operations and enable users to make requests from the front end to the services on the back end, we created a RESTful API with Django to handle data requests and responses between the front end and the Firebase real-time database. To return JSON data to React, we used the Django REST framework which is a Django extension that includes tools and utilities for creating RESTful APIs. We also used Python to create three microservices which are the *Training Builder Service* (TBS), the *Training Tips Service* (TTS), and the *Training Fetch Service* (TFS). The functions and classes are all made available to the API. Figure 1 shows an overview of REIT-Builder.

In detail, at the creation of a new training program, the user needs to provide basic information about the new training (name, description, resources, size class). Once created, the training program can be populated by selecting and customizing activities from the available groups (lecture, interview, review, assessment, reflection, other). The initial set of pre-defined activities is extracted from SaPeer and ReverseSaPeer [12] and based on active and experiential learning best practices. Upon selecting the activity, an API call is made and routed to the TBS. This service is responsible for adding an activity, inserting an activity, deleting an activity, storing a training program, and requesting recommended tips from the TTS. When the service is requested, the service checks if the activity is valid, make modifications, and fetch recommendation from the TTS. After each activity is added and checked by the TBS, the specific activity is shown to the user and all subsequent activities are added following the same procedure to display the pattern of the training. The TBS saves the customized training to the database located in Firebase. This process is accomplished by leveraging the Django Rest framework to make API calls and save data to the database.

Upon successfully saving the training into the database, a cron job in the TFS runs to get all the training programs from Firebase and make them available to the API which is accessible by the React application to display all existing training programs currently present in the database. The user can always access existing training programs when navigating the web application and see the corresponding activities for each training.

## 3. Use Cases

Once arrived on the landing page (Figure 2.(a)), users can start working on their program by clicking “get started” at the top of the page. Together with the option of creating a new program, REIT-Builder offers the possibility of reusing an existing training program (Figure 2.(b)). Once selected, the program is displayed as a sequence of color-coded activities with all the necessary information needed to execute the program (Figure 2.(c)).



Build custom trainings and discover existing trainings from a large variety of professors just like you.

Build a custom workflow filled with activities and tailored to your needs.

Control the time of activities estimated for the training.

Get recommendations while customizing your training.

Expect more, design your trainingsfor more productivity, and more value.

SaPeer uses role-playing to enable students to experience first-hand the interviewing process.

testing description beboibskf slkbfskljf lkjbfkljb rsklgbskjbsk lksbfsklbfskldfl lskdfjbslkfbslk fslkbsdkljbsdl flkbflkjbgqb lkjbgflksbglkfsj

(a) (b) (c)

**Figure 2:** (a) REIT-Builder landing page; (b) Existing training programs; (c) Visualization of a program.

If the user decides to create a new training program, the user is asked to provide (1) a name for the program, (2) a description, (3) the available time of the teaching team for this program, and (4) the number of students in the class (Figure 3.(a)). Upon successfully giving the parameters, users are prompted with the next step to start building a custom training, where they can select the activity they want to perform (Figure 3.(b)). During the creation of a program, the REIT-Builder provides tips on the next activity to include or exclude and offers guidance on the utilization of resources (Figure 3.(c)).

Different activities have different options and available resources. For example, when a lecture is selected, the users have the choice between live or recorded lectures and, in the latter case, existing resources are provided (Figure 3.(d)). In the case of interviews, users can choose between watching an existing interview, and thus uploading a link where the interview is available (Figure 3.(e)), or having their students perform an interview. In that case, the user needs to customize the activities in terms of length and assigning a role to the student (Figure 3.(f)). Analogously, the self-assessment activity allows one to either use existing resources or define new ones (Figure 3.(g))

All the available interactions between the user and REIT-Builder are represented in the use case in Figure 4.

## 4. Evaluation Plan

The current version of REIT-Builder enables users to input new training programs and view existing ones. The system supports the users by verifying that the chosen or built program is suitable for the size of the class and the available resources. In addition, REIT-Builder gives recommendations on how to build the training that can guide users in the process.

To evaluate the current implementation of REIT-Builder and collect data to inform its evolution, we are planning a user study with a set of RE instructors and young professionals. The study consists of two parts. In the first part, the focus is on evaluating the usability of the tool. So, participants will be given a script to build a training program with specific activities and required modifications. During their execution, we will collect traditional usability data such as the time of execution and the number of mistakes they made. At the end of the study, we will gather feedback from participants regarding their experience using the tool, as well as their perceptions of the benefits and challenges associated with the approach. The focus of the

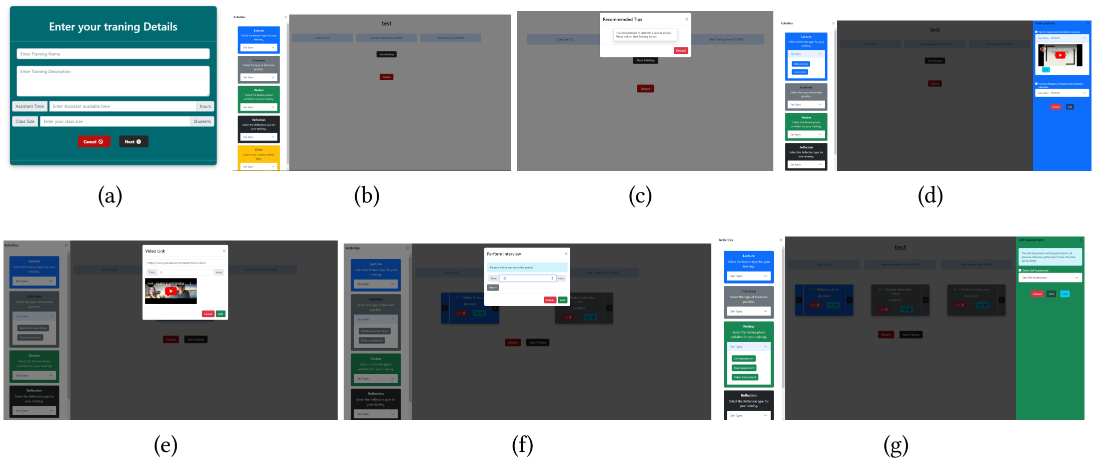


**Figure 3:** (a) Starting page for the creation of a new training program; (b) Available activities to select; (c) Provided tip; (d) Options available for lectures; (e) Option to add a video of an interview; (f) Customization of an interview; (g) Self-assessment activity.

second part of the study is to evaluate the contribution of the tips and how they are perceived by instructors. To this aim, participants will be asked to create a training that fits their class's and team's needs. During the study, we will measure how often tips are followed and how many warnings about violations of the resources REIT-Builder created, and at the end, we will collect feedback on the perceived usefulness of the received tips and the satisfaction with respect to the built program.

## 5. Conclusion and Future Work

This tool paper presents the initial implementation of REIT-Builder, a tool to support requirements analysts in academia and industry to create effective training programs for requirements elicitation interviews suitable for the user's resources and needs. REIT-Builder is built using resources and best practices from the literature. As the next steps, we plan to first expand the set of tips that REIT-Builder is able to provide, and then evolve REIT-Builder into a recommendation engine able to propose a customized list of training that fits the user requirements without the need for the user to manually select activities.

## Acknowledgments

This work was partially supported by the National Science Foundation under grant CCF-1718377.

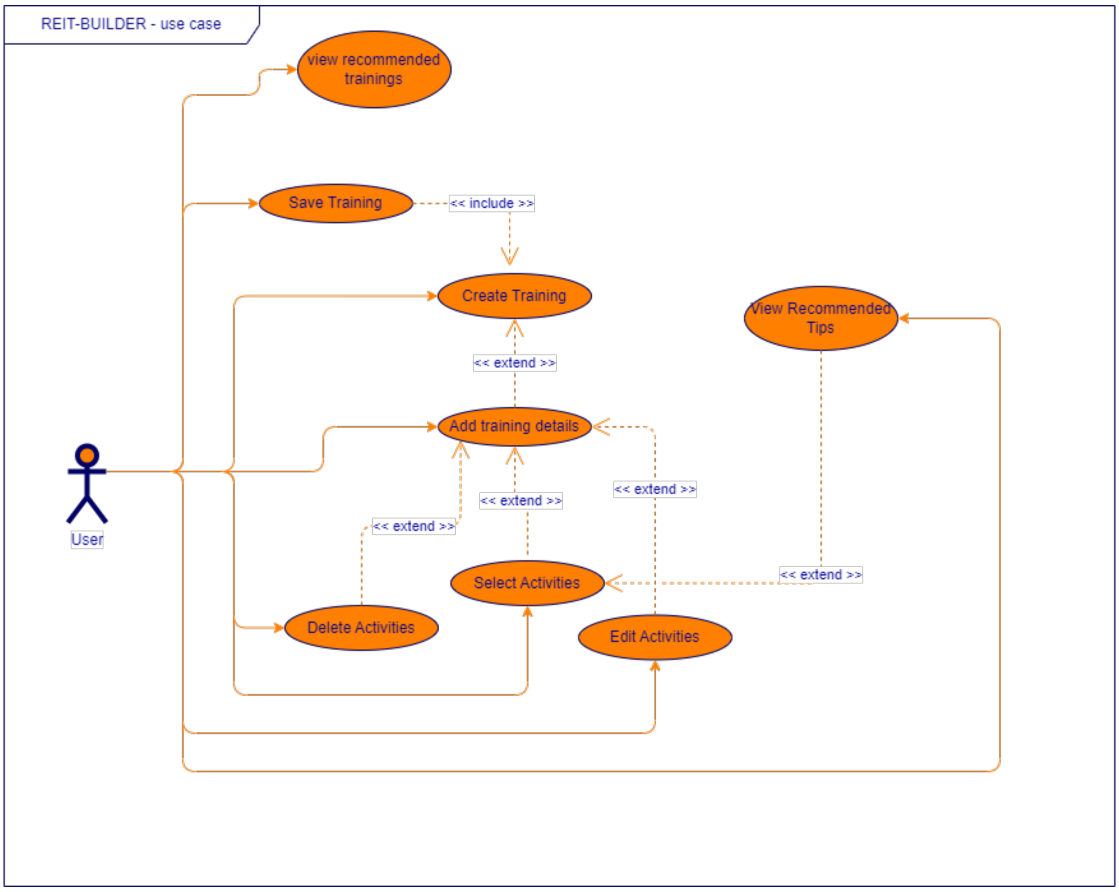


**Figure 4:** Use Case Diagram of REIT-Builder.

## References

[1] F. García-Peñalvo, H. Alarcon, A. Dominguez, Active learning experiences in engineering education, International Journal of Engineering Education 35 (2018) 305–309.

[2] S. B. Bruce R Maxim, Sushil Acharya, M. Kessentini, Wip: Introducing active learning in a software engineering course, in: 2017 ASEE Annual Conference & Exposition, 10.18260/1-2–29132, ASEE Conferences, 2017.

[3] B. R. Maxim, A. Decker, J. J. Yackley, Student engagement in active learning software engineering courses, in: 2019 IEEE Frontiers in Education Conference (FIE), 2019, pp. 1–5. doi:10.1109/FIE43999.2019.9028644.

[4] E. D. Ragan, S. Frezza, J. Cannell, Product-based learning in software engineering education, in: 2009 39th IEEE Frontiers in Education Conference, 2009, pp. 1–6. doi:10.1109/FIE.2009.5350648.

[5] M. Daun, A. Grubb, V. Stenkova, B. Tenbergen, A systematic literature review of requirements engineering education, Requirements Engineering (2022).

[6] L. P. Álvarez Reyes, B. Cuesta Quintero, Teaching based on models and transformations under the active learning approach, Journal of Physics: Conference Series 1513 (2020). doi:10.1088/1742-6596/1513/1/012012.

[7] I. Hadar, P. Soffer, K. Kenzi, The role of domain knowledge in requirements elicitation via interviews: An exploratory study, Requirements Engineering 19 (2014) 143–159. doi:10.1007/s00766-012-0163-2.

[8] D. Zowghi, C. Coulin, Requirements elicitation: A survey of techniques, approaches, and tools, Springer Berlin Heidelberg, Berlin, Heidelberg, 2005, pp. 19–46. doi:10.1007/3-540-28244-0_2.

[9] B. Donati, A. Ferrari, P. Spoletini, S. Gnesi, Common mistakes of student analysts in requirements elicitation interviews, in: International Working Conference on Requirements Engineering: Foundation for Software Quality, Springer, 2017, pp. 148–164.

[10] M. G. Pitts, G. J. Browne, Improving requirements elicitation: an empirical investigation of procedural prompts, Information systems journal 17 (2007) 89–110.

[11] M. Maher, N. Dehbozorgi, M. Dorodchi, S. Macneil, Design patterns for active learning, Faculty Experiences in Active Learning: A Collection of Strategies for Implementing Active Learning Across Disciplines (2020).

[12] A. Ferrari, P. Spoletini, M. Bano, D. Zowghi, Sapeer and reversesapeer: teaching requirements elicitation interviews with role-playing and role reversal, Requirements Engineering 25 (2020) 417–438.

[13] J. Vilela, A. Ferrari, Sapeer approach for training requirements analysts: An application tailored to a low-resource context, Springer-Verlag, 2021.

[14] Facebook, ReactJS, https://reactjs.org/, 2013. Accessed: February 18, 2023.

[15] Django Software Foundation, Django, https://www.djangoproject.com/, 2021. Accessed: February 18, 2023.

[16] Google, Firebase, https://firebase.google.com/, 2011. Accessed: February 18, 2023.