TALLINN UNIVERSITY OF TECHNOLOGY
School of Information Technologies

Anton Antonov 185376IAIB

Vladislav Poljakov 185104IAIB

German Hanmamedov 185333IAIB

Onboarding System for TalTech Courses and Curricula

Bachelor's thesis

Supervisor: Ago Luberg
PhD

Tallinn 2022

TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond

Anton Antonov 185376IAIB

Vladislav Poljakov 185104IAIB

German Hanmamedov 185333IAIB

Sisseelamissüsteem TalTechi kursustele ja õppekavadele

Bakalaureusetöö

Juhendaja: Ago Luberg
PhD

Tallinn 2022

Author’s declaration of originality

We hereby certify that we are the sole authors of this thesis. All the used materials, references to the literature and the work of others have been referred to. This thesis has not been presented for examination anywhere else.

Authors: Anton Antonov, Vladislav Poljakov, German Hanmamedov

30.05.2022

Abstract

The purpose of this bachelor thesis is to develop a system for Tallinn University of Technology, which will allow lecturers to create web pages with questionnaires to show students what to expect in this course or study programme. Such onboardings might help lecturers collect feedback, such as whether students find the study difficult or not and what can help give the first lecture more effectively. Also, according to the statistics, the lecturer can improve the course or add any additional helping materials for students that might decrease the dropout rate. The service supports various types of users with different functionality. The lecturer will be able to create, modify and delete the onboarding and check results and visualized graphs. Students will only be able to complete questionnaires without the possibility to create their own or modify an existing one. Users can log in with their Uni-ID so that the service identifies them as a lecturer or a student. Guests will only be able to complete an open onboarding if the lecturer allows it.

This project is built following a Client-Server architecture and divided into two parts: backend as a RESTful API and frontend as a Web application.

The thesis is in English and contains 37 pages of text, 7 chapters, 18 figures.

Annotatsioon

Käesoleva bakalaureusetöö eesmärk on luua Tallinna Tehnikaülikoolile süsteem, mis annab õppejõule võimaluse koostada kursust või õppekava tutvustav veebileht koos tagasisideküsimustega. Külastaja saab infot, mida kursus või õppekava endast kujutab, ühtlasi saab ülikool külastajalt tagasisidet. Näiteks on võimalik koguda tekkinud küsimusi, saada aimu, kui keeruliseks õppija kursust või õppekava peab, või saada teada võimalikest õppimist segavatest takistustest. Teenus toetab eritüüpi kasutajaid erineva funktsionaalsusega. Õppejõud saavad luua, muuta ja tühistada loodud tutvustuslehti ning kontrollida statistikat. Üliõpilased saavad tutvustusi vaadata ja vastata küsimustele. Üliõpilased saavad sisse logida oma Uni-ID kaudu, et teenus tuvastaks neid tudengina. Kui õppejõud on vastava loa andnud, saavad tutvustusi vaadata ka külalised.

See projekt on ehitatud vastavalt kliendi-serveri arhitektuurile ja jagatud kaheks osaks: serverirakendus (REST API) ja kasutajaliides (veebirakendus).

Lõputöö on kirjutatud inglise keeles ning sisaldab teksti 37 leheküljel, 7 peatükki, 18 joonist.

List of abbreviations and terms

API	Application Programming Interface
CD	Continuous Delivery
CI	Continuous Integration
CSS	Cascading Style Sheets
HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol
JSON	JavaScript object Notation, data-interchange format
Onboarding	A web page with questionnaires to introduce the course to students and collect statistics.
REST	Representational State Transfer, software architectural style
SPA	Single Page Application
UI	User Interface
URL	Uniform Resource Locator
UX	User Experience

Table of contents

List of figures

1. Introduction

1.1 Background

At this time, the first lectures at the university are spent introducing the course to the new students and collecting their questions and opinions, but it is hard to gather everyone's thoughts at the first meeting. To do this, lecturers can create onboarding pages with questionnaires such as Google Forms or Microsoft Forms to get feedback from students and make the first lecture more effective and decrease the dropout rate from the course. However, these services do not suit the current project purposes due to the factors that will be described further in the section “6.1.4 Searching for analogs”.

1.2 Purpose

Our goal in this thesis is to create a service for TalTech University which could provide a convenient substitute for the analogs to develop questionnaires. It should be easy to use for those who are not familiar with IT. And also have distinct functionality and structure which could be suitable for the university (for example, the separation by areas).
The questionnaires must be in onboarding format, which means that they can contain a lot of educational information such as text, video, pictures, and question blocks. Also, it should be possible to use the service to create information pages.
The service should support the possibility of dividing the questionnaires by study areas so that the questionnaires of the IT area will be stored separately from the questionnaires of Economics, etc.
The results of the questionnaires should be stored and visualized using graphs to provide the statistics.

1.3 Requirements

- The service should have the possibility to separate onboarding by study area. Lecturers will only be able to create onboardings in the areas they belong to.
- The service should provide the possibility to add text blocks, pictures, videos, and questions to the onboarding.

- Graphs should be created based on the results, from which lecturers have an easy way to check the statistics.
- It should be possible to create, change and delete onboardings.
- It should be possible to see a preview of the onboarding.
- It should be possible to share a link to the onboarding.
- The service must support 3 types of users: lecturer, student, and guest.
- Students can only pass onboardings in their study area.
- Guests can only access the public onboardings.
- It should be possible to log in via Uni-ID.
- The process of creating onboarding should be understandable even to a person not connected with IT.

2. Project Design

This project is split into 3 parts: database, frontend (SPA web application) and backend (REST service).

2.1 Database

The database in our project is used only for storing the answers of the users. For every onboarding its own table is created with the first response (initially onboarding doesn't have one). As there is an undefined number of questions, the table is created in a loop (`code reference: src/back-end/app.js 273:345`). The name of every column is the question itself. As the table can't contain multiple columns with the same name, "$%#n" is added to each of them (example: New question$%#0 is the first one, New question$%#1 is the second one). There are 3 types of questions at the moment: radio, checkbox, and text input. For storing checkbox values "*&%$#@" separator is used, to allow using special characters in the answer, because checkbox answers are being stored as a String and it is needed to split them. Also, for marking text answers symbol combination "*&@" is used to exclude them from the dashboard view, as all the answers are different, and there is no need to show them as graphs. Instead, it is possible to check these answers in the Results view.

2.1.1 PostgreSQL

We are using the PostgreSQL database in our project because we wanted to learn something new, but the main reason was that it is free. PostgreSQL is a powerful, open source object-relational database system with over 30 years of active development that has earned it a strong reputation for reliability, feature robustness, and performance. [1]

2.2 Backend

RESTful API service in our project is built using JavaScript with Node.js [2] framework. Representational state transfer (REST) is a software architectural style that was created to guide the design and development of the architecture for the World Wide Web. REST defines a set of constraints for how the architecture of an Internet-scale distributed hypermedia system, such as the Web, should behave. The REST architectural style emphasizes the scalability of interactions between components, uniform interfaces, independent deployment of components, and the creation of a layered architecture to facilitate caching components to reduce user-perceived latency, enforce security, and encapsulate legacy systems. [3] The visualization of the RESTful API is shown in Figure 1.

In some requests queries are generated to save the results of the onboarding. This data is being retrieved with the SELECT clause and is sent to the frontend as ready-to-use data for charts in order to visualize the answers.

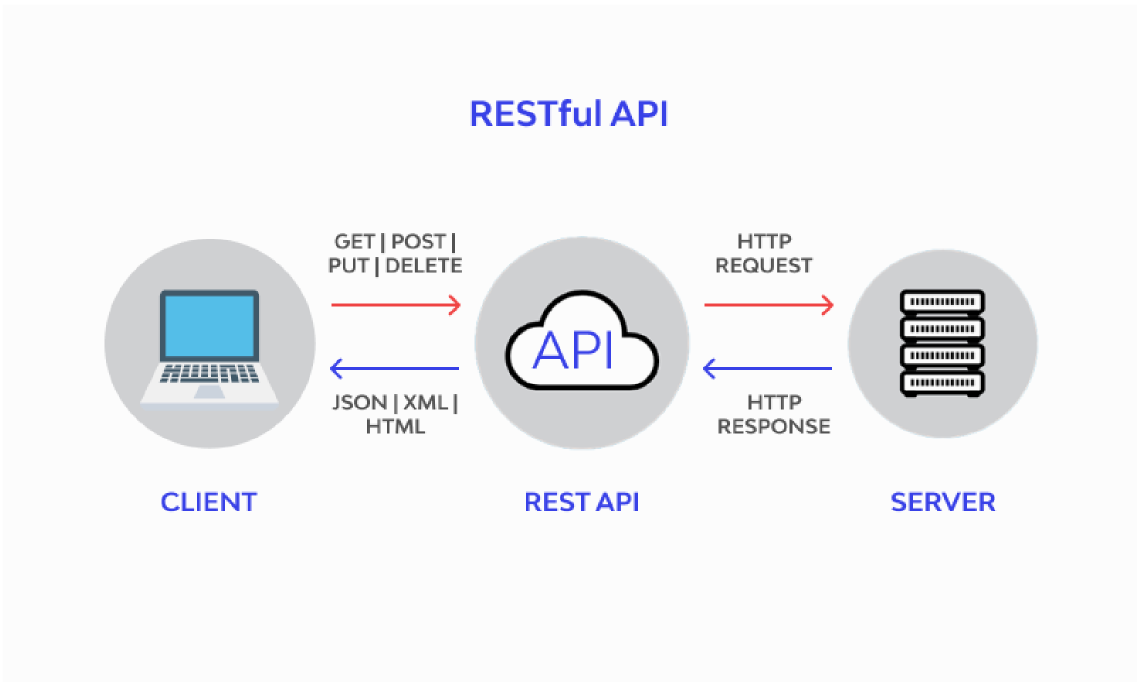


Figure 1. REST API

2.3 Frontend

Frontend is done as a single-page application (SPA) using React JS. [4] Initially we were choosing between Angular and React and eventually chose the second one, because firstly, new knowledge is always a good thing and secondly, React has some advantages over Angular, such as simplicity and convenience. Also, React has a lot of libraries that are easy to install and integrate.

SPA is a web application or website that interacts with the user by dynamically rewriting the current web page with new data from the web server, instead of the default method of a web browser loading entire new pages. The goal is faster transitions that make the website feel more like a native app.

In a SPA, a page refresh never occurs; instead, all necessary HTML, JavaScript, and CSS code is either retrieved by the browser with a single page load, or the appropriate resources are dynamically loaded and added to the page as necessary, usually in response to user actions. [5]

2.4 Backend and frontend communication

Backend communicates with the frontend using API via HTTP protocol. Data is transferred between backend and frontend in JSON format. For exchanging data a HTTP client called Axios was added. Axios is a simple promise based HTTP client for the browser and node.js. Axios provides a simple to use library in a small package with a very extensible interface. [6]

3. Development process

The development process was made in the GitLab environment. [7] The project includes 3 parts: the frontend, backend, and database. The frontend and the backend were put into separate repositories to make the deployment more convenient. The backend repository contains Node.js REST API and PostgreSQL database. The development and the testing stages were made locally. For the deployment, every part of the project was put into the docker container, and a docker image was made.

3.1 Frontend setup

The web application is developed in React framework. To run the frontend locally, Node.js and npm [8] must be installed on the computer. Before starting the project, all necessary dependencies must be installed by the command `npm install`. The application starts working after the console command `npm start`.

3.1.1 Connection to the REST API

The web application connects with the REST API through the URL address specified in the code, depending on the needed endpoint. To send and receive data, the Axios library was used.

3.2 Used libraries

3.2.1 React player

It is a React component for playing a variety of URLs, including file paths, YouTube, Facebook, Twitch, SoundCloud, Streamable, Vimeo, Wistia, Mixcloud, DailyMotion and Kaltura. [9] It is used in the project to add videos to the onboardings by providing the URL from one of the above listed resources.

3.2.2 Universal cookies

This library was used in the project to apply cookies for storing data related to passing the onboardings. [10] HTTP cookies (also called web cookies, Internet cookies, browser cookies, or simply cookies) are small blocks of data created by a web server while a user is browsing a website and placed on the user's computer or other device by the user's web browser. [11]

3.2.3 React beautiful drag and drop

This React component is used in our project to drag and drop the questions to change their order in the onboarding. react-beautiful-dnd is a higher level abstraction specifically built for lists (vertical, horizontal, movement between lists, nested lists and so on). Within that subset of functionality react-beautiful-dnd offers a powerful, natural and beautiful drag and drop experience. [12]

3.2.4 React icons

This library contains a lot of different icons, which can be easily integrated into the project. Most icons in the project were taken from it. [13] The example is shown in the Figure 2.

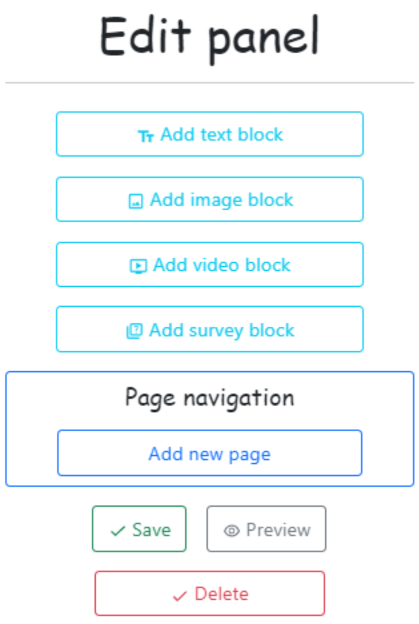
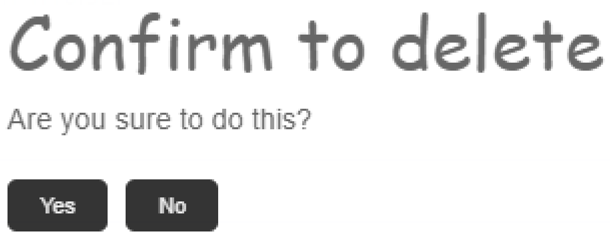


Figure 2. Edit panel

3.2.5 React confirm and custom alert

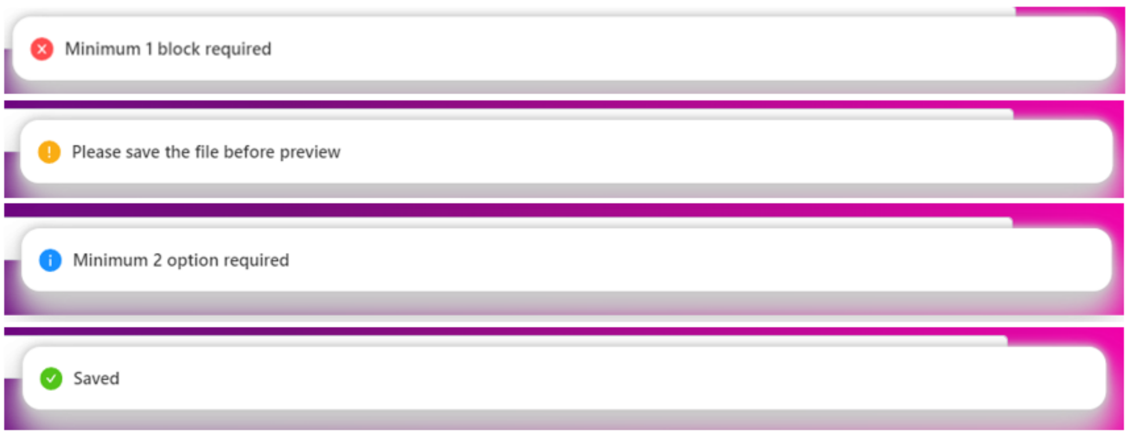
react-confirm-alert component was used as a confirmation of deletion of the onboarding. [14] An example of confirm alert is shown in Figure 3.


Confirm to delete

Are you sure to do this?

Figure 3. React confirm alert example

To provide notifications about actions that were done on the page, a react-custom-alert component was used. [15] Examples of custom alerts in this project are shown in Figure 4.


Minimum 1 block required

Please save the file before preview

Minimum 2 option required

Saved

Figure 4. React custom alert examples

3.2.6 React UUID

A UUID (Universal Unique Identifier) is a 128-bit number used to uniquely identify some object or entity on the Internet. Depending on the specific mechanisms used, a UUID is either guaranteed to be different or is, at least, extremely likely to be different from any other UUID generated until 3400 A.D. In its authoritative textual representation, the 16 octets of a UUID are represented as 32 hexadecimal (base-16) digits, displayed in 5 groups separated by hyphens, in the form 8-4-4-4-12 for a total of 36 characters (32 alphanumeric characters and 4 hyphens). [16] This component was used to generate UUIDs for every onboarding to unify their name so that there were no name conflicts in the JSON file (where onboarding itself is stored) and in the database, where each table name is onboarding’s UUID.

3.2.7 React router dom

React Router is a lightweight, fully-featured routing library for the React JavaScript library. React Router runs everywhere that React runs; on the web, on the server (using node.js), and on React Native. [17] This component is used to redirect between pages.

3.2.8 MUI

Material UI is a comprehensive library of components that features implementation of Google's Material Design system. [18] In the project, it was used to design buttons, sidebar, selectors, etc.

3.2.9 Chart.js

Chart.js is a React framework to create different types of charts. This component was used to visualize results from onboardings by graphs. Bar, pie, polar area, and doughnut charts are used in the project. It is possible to select between them in the Dashboard view. [19]

3.3 Backend setup

The RESTful API service is developed in Node.js. The backend also needs Node.js and npm to be installed on the computer to run the service locally. Dependencies downloading for the backend is also necessary and can be done by the command `npm install`. The backend will be started after entering the `node app.js` command into the console from the backend root folder.

3.4 Database setup

As a database, we are using PostgreSQL. It is used to store all records made by completing an onboarding. The current database doesn’t have any predefined tables. Tables are being created after submitting the first response to the onboarding. Database properties are located in the backend `app.js` file.

3.5 Docker

Every part of the project was put into the docker containers and deployed on the server that was provided by the supervisor. Docker allows packaging applications into containers, which are standardized executable components. These components contain application source code, operating system libraries, and required dependencies. Using Docker containers allows running the application in any environment, which supports Docker.

For both frontend and backend Docker files were made, which are located in the root folders and named `Dockerfile`. Also, in the project root folder, there is a `docker-compose.yml` file, which allows running a multi-container Docker application. It requires Docker preinstalled on the computer or the server and can be executed by the command `docker-compose up -d`.

3.6 Gitlab CI/CD

For the deployment, a GitLab CI/CD tool was used. This tool allows creating pipelines to run predefined jobs such as building, testing, putting the application into Docker containers, and deploying to the server. On the server side, GitLab Runners are installed and are connected to the project repositories. In each repository, there is a `gitlab-ci.yml` file where all stages and jobs are predefined. This file creates pipelines when the code in the main branch was changed.

3.7 Testing

3.7.1 Backend testing

At the current stage of development, automated tests were not implemented and configured. All features are tested manually.

3.7.2 Frontend testing

At the current stage of development, automated tests were not implemented and configured. All features are tested manually.

4. Project description

The project is easy to use, both on the student side and on the lecturer side as well. This service is a partially public web application. That means that each guest user can go to the web application page. The service supports different types of roles. Users will have different restrictions depending on their role. The project is an environment for TalTech university teachers to create informative pages that will teach students and collect statistical data, if needed. The website is easy to understand and has no complicated processes, so that even a non-IT person can use it without any problems. Unlike its analogues, this service has a functionality, structure and design that is aimed at the Onboarding questionnaire format, which will not only collect responses from students, but will also contain quite a lot of educational information.

4.1 Roles

The service supports three different types of roles: Lecturer, Student, Guest. The main functions available for users according to the role are listed below.

4.1.1 Lecturer

The lecturer is a user who authenticated through the Uni-ID of the teacher from TalTech University. Based on the authentication, we can check what is the study area of this particular lecturer and set restrictions, so the lecturer can create/delete/update only onboardings related to his study area.

Available features for lecturer, at this stage of development, are:

- Create a new onboarding page with the possibility to add content.
- Edit existing onboarding (move, copy, add, remove content blocks; add pages; change page title; change onboarding name).
- Delete all data related to the onboarding (JSON file, images, database table).
- Preview of the onboarding. This function is used to overview the onboarding without the submit option.
- Onboarding sharing. Onboarding is available via the link, and results can be submitted.
- Overview of results with all recorded answers for the particular onboarding and their visualization via graphs.

4.1.2 Student

A student is an authenticated user. This service supports only authentication through the Uni-ID, so if the user successfully passed it, then he is obviously a student (if not a lecturer). Students are allowed only to complete open (available for everyone) and closed (available only for authenticated users) onboardings.

4.1.3 Guest

A guest is a non-authenticated user. Guests can pass only onboardings that are open to them by a lecturer. If the lecturer didn’t specify in onboarding settings that this one is available both for students and guests, then it means that this onboarding is available only for authenticated users (students).

4.2 UX/UI

This service has a responsive user interface and is available on different types of devices, besides the mobile phone. For modern UI Bootstrap styles were used. Bootstrap is a free and open-source CSS framework directed at responsive, mobile-first front-end web development. It contains HTML, CSS and (optionally) JavaScript-based design templates for typography, forms, buttons, navigation, and other interface components. [20]

5. Service functionality

Due to problems that have occurred, which will be described in more detail in the "Discussion" part of this document, some of the planned functionality has not yet been completed. In this section, will be described the functionality that has been developed so far on each page.

5.1 Login page

A simple login page has been implemented so that the creation of the onboarding would not be available for everyone.

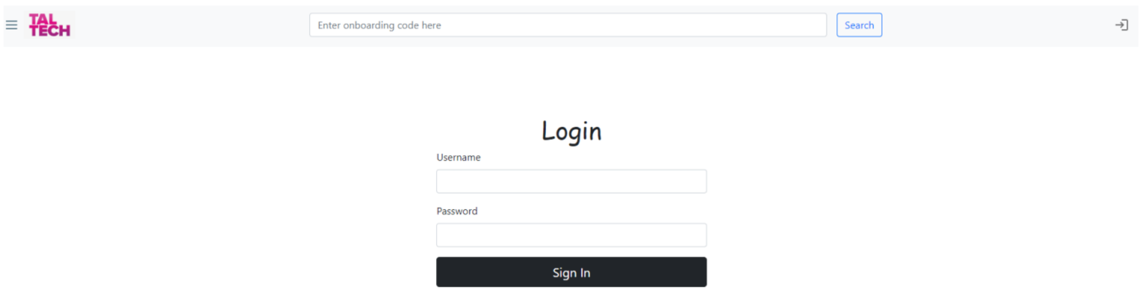


Figure 5. Login page

5.2 Main page

When logged in, the page will be redirected to the main page with the study area choice. After moving the cursor to the study area "Create onboarding" and "All onboardings" buttons will appear. (Figure 6)

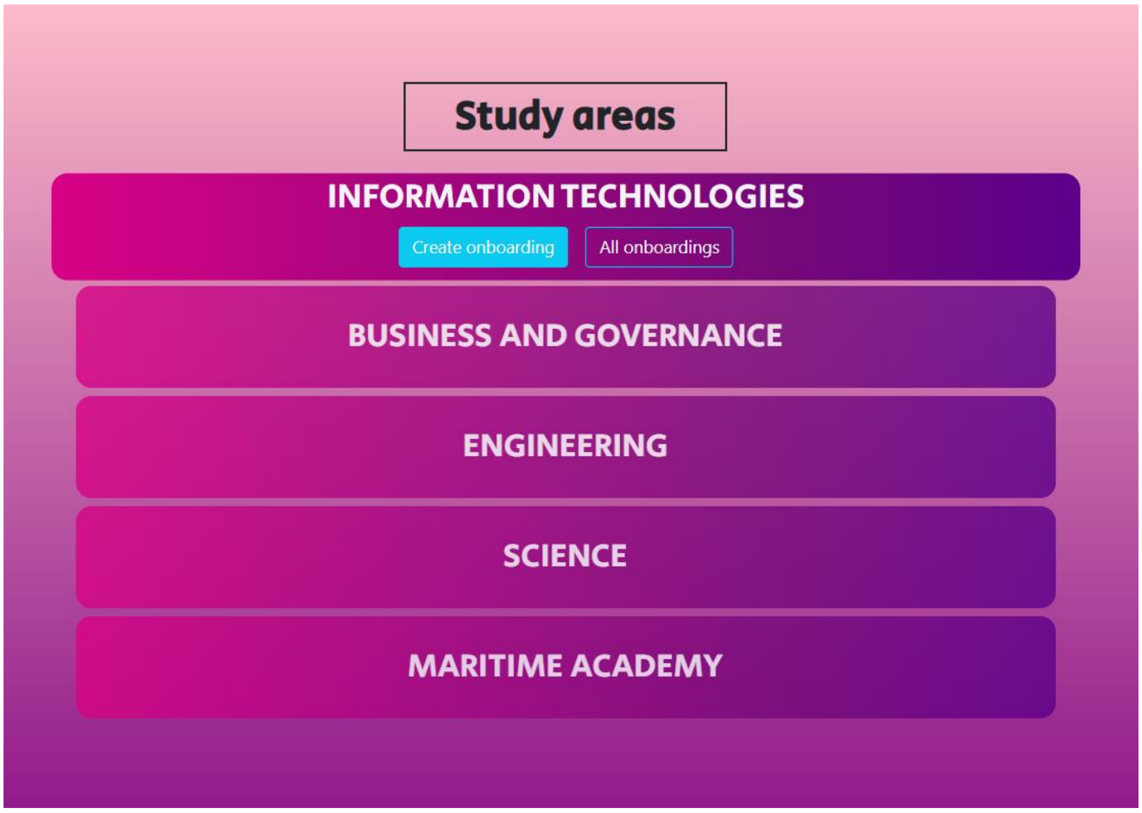


Figure 6. Study areas view

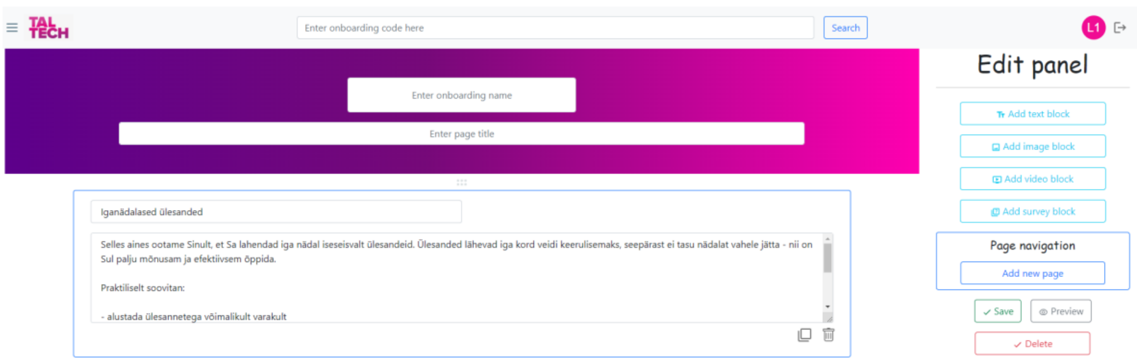
5.3 Onboarding creation and editing

To create or change onboarding, you will be moved to the same page, the only difference will be the URL of the site. (Figure 7)

For creating → {study area}/create

For editing → {study area}/edit

To avoid using complex URLs, such information as the unique onboarding ID is handled through the cookies.



Iganädalased ülesanded

Selles aines ootame Sinult, et Sa lahendad iga nädal iseseisvalt ülesandeid. Ülesanded lähevad iga kord veidi keerulisemaks, seepärast ei tasu nädalat vahele jätta - nii on Sul palju mõnusam ja efektiivsem õppida.

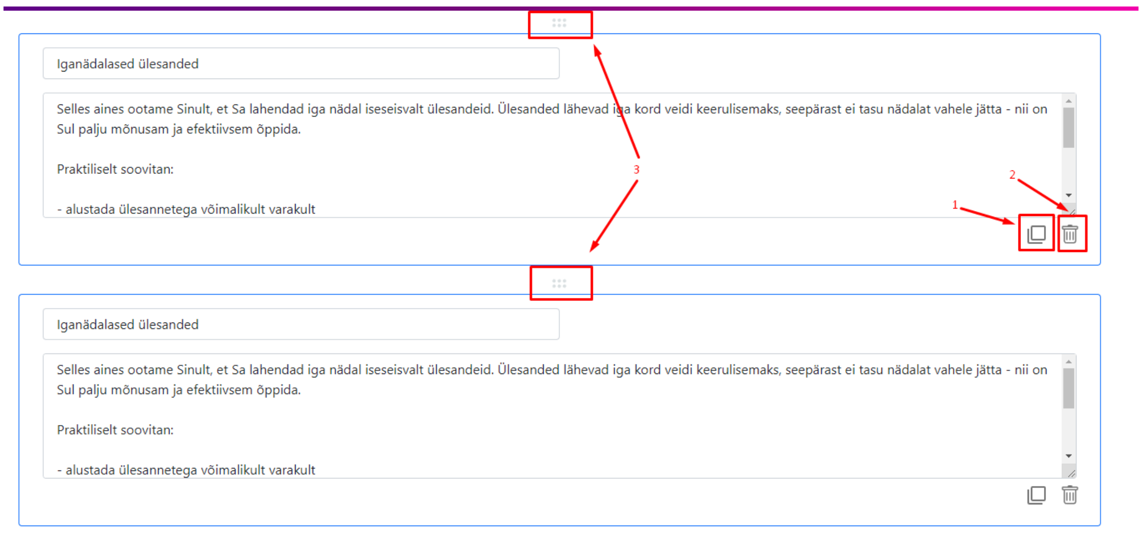
Praktiliselt soovitan:

- alustada ülesannetega võimalikult varakult

Figure 7. Edit/Create page view

On this particular page, it is possible to:

- Specify the name of the onboarding (must be unique in the study area)
- Specify page title
- Use the edit panel to edit onboarding’s content.
- Edit panel has the following functionality:
    - ➔ Add text block
    - ➔ Add image block
    - ➔ Add video block
    - ➔ Add survey block (single/multiple choice, text input)
    - ➔ Add new page
- 1 → Copy the block, 2 → Delete the block, 3 → Drag the block (Figure 8)



Iganädalased ülesanded

Selles aines ootame Sinult, et Sa lahendad iga nädal iseseisvalt ülesandeid. Ülesanded lähevad iga kord veidi keerulisemaks, seepärast ei tasu nädalat vahele jätta - nii on Sul palju mõnusam ja efektiivsem õppida.

Praktiliselt soovitan:

- alustada ülesannetega võimalikult varakult

Iganädalased ülesanded

Selles aines ootame Sinult, et Sa lahendad iga nädal iseseisvalt ülesandeid. Ülesanded lähevad iga kord veidi keerulisemaks, seepärast ei tasu nädalat vahele jätta - nii on Sul palju mõnusam ja efektiivsem õppida.

Praktiliselt soovitan:

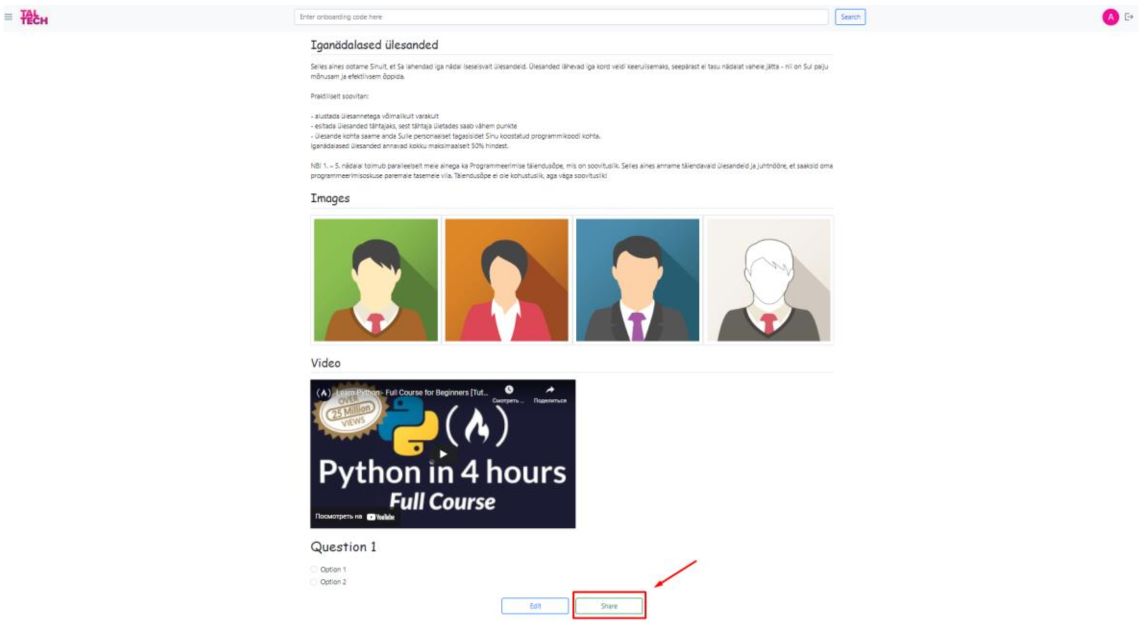
- alustada ülesannetega võimalikult varakult

Figure 8. Functionality of the blocks

- Save (store the onboarding to the backend in JSON format)
- Preview (check the student's view)
- Delete (delete this particular onboarding with all related information from the database)

5.4 Onboarding preview

Lecturers can pre-check the onboarding view for students in advance using the Preview button. For sharing particular onboarding to others, lecturers need to use the Share button. (Figure 9)



Iganädalased ülesanded

Selles aines ootame Sinult, et Sa lahendad iga nädal iseseisvalt ülesandeid. Ülesanded lähevad iga kord veidi keerulisemaks, seepärast ei tasu nädalat vahele jätta - nii on Sul palju mõnusam ja efektiivsem õppida.

Praktiliselt soovitan:

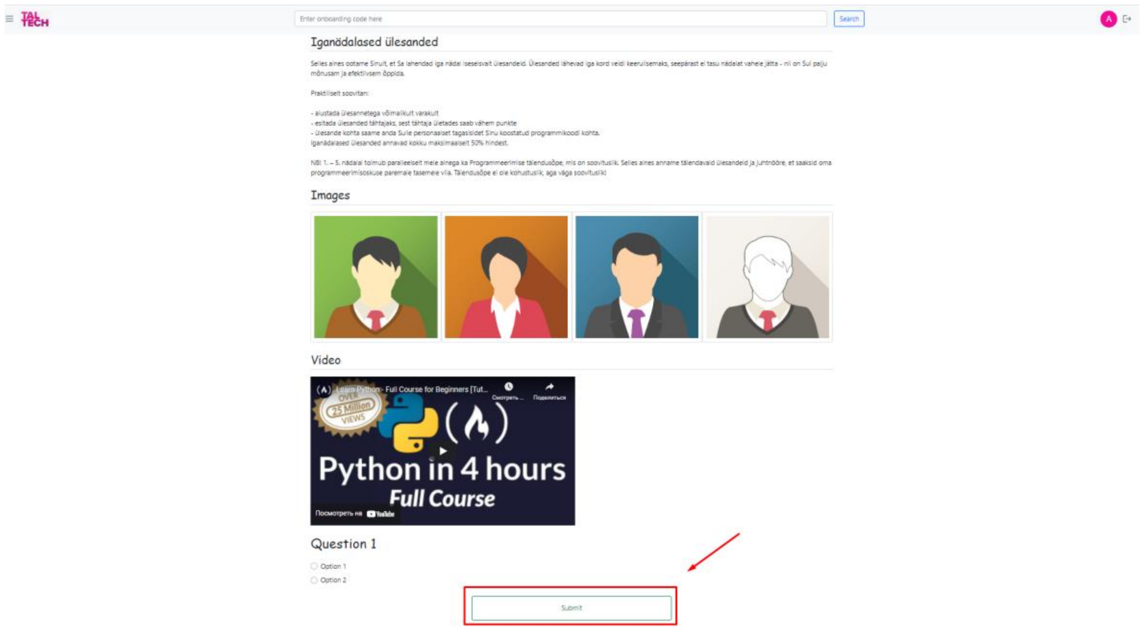
- alustada ülesannetega võimalikult varakult
- esitada ülesanded tähtajaks, sest tähtaja ületades saab vähem punkte
- ülesande kohta saame anda Sulle personaalset tagasisidet Sinu koostatud programmikoodi kohta.
Iganädalased ülesanded annavad kokku maksimaalselt 50% hindest.

NB! 1. – 5. nädalal toimub paralleelselt meie ainega ka Programmeerimise täiendusõpe, mis on soovituslik. Selles aines anname täiendavaid ülesandeid ja juhtnööre, et saaksid oma programmeerimisoskuse paremale tasemele viia. Täiendusõpe ei ole kohustuslik, aga väga soovituslik!

Figure 9. Preview view

5.5 Onboarding sharing

Clicking on the Share button will redirect the lecturer to the onboarding view for students. Lecturers can use the URL from this page to share with others. Submission of the onboarding is only available from this view. On the last page of the onboarding, it is possible to submit the answers if all fields were filled. (Figure 10)



Enter onboarding code here

Search

Iganädalased ülesanded

Selles aines ootame Sinult, et Sa lahendad iga nädal iseseisvalt ülesandeid. Ülesanded lähevad iga kord veidi keerulisemaks, seepärast ei tasu nädalat vahele jätta - nii on Sul palju mõnusam ja efektiivsem õppida.

Praktiliselt soovitan:

- alustada ülesannetega võimalikult varakult
- esitada ülesanded tähtajaks, sest tähtaja ületades saab vähem punkte
- ülesande kohta saame anda Sulle personaalset tagasisidet Sinu koostatud programmikoodi kohta.
Iganädalased ülesanded annavad kokku maksimaalselt 50% hindest.

NB! 1. – 5. nädalal toimub paralleelselt meie ainega ka Programmeerimise täiendusõpe, mis on soovituslik. Selles aines anname täiendavaid ülesandeid ja juhtnööre, et saaksid oma programmeerimisoskuse paremale tasemele viia. Täiendusõpe ei ole kohustuslik, aga väga soovituslik!

Images

Video

Question 1

Option 1

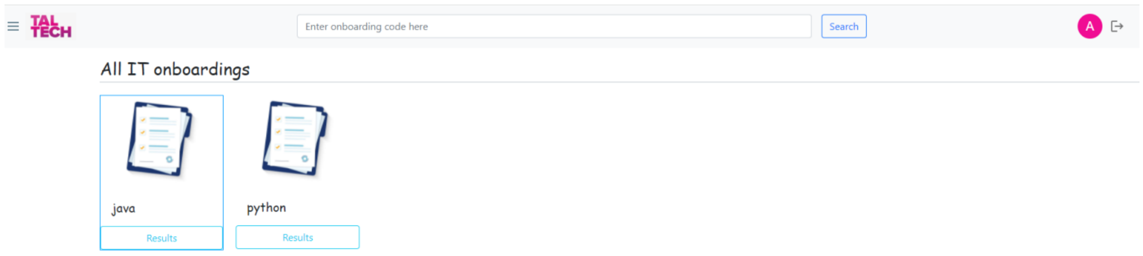
Option 2

Submit

Figure 10. Share view

5.6 All onboardings

This page contains all onboardings of the current study area (which is saved to cookies) that the lecturer chose before. (Figure 11)



Enter onboarding code here

Search

All IT onboardings

java

Results

python

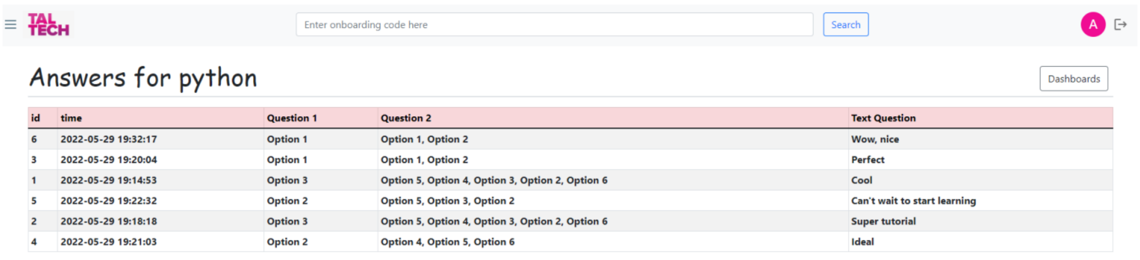
Results

Figure 11. All onboardings view

Onboardings have a Results button. By clicking on it, the lecturer can check the provided answers.

5.7 Results

If onboarding has answers, then a table with the results will appear. This table is basically the visualization of the onboarding's database table. This table shows the answers for each question and the time of submission. If no answers were provided, the corresponding message will be shown.



Answers for python

id	time	Question 1	Question 2	Text Question
6	2022-05-29 19:32:17	Option 1	Option 1, Option 2	Wow, nice
3	2022-05-29 19:20:04	Option 1	Option 1, Option 2	Perfect
1	2022-05-29 19:14:53	Option 3	Option 5, Option 4, Option 3, Option 2, Option 6	Cool
5	2022-05-29 19:22:32	Option 2	Option 5, Option 3, Option 2	Can't wait to start learning
2	2022-05-29 19:18:18	Option 3	Option 5, Option 4, Option 3, Option 2, Option 6	Super tutorial
4	2022-05-29 19:21:03	Option 2	Option 4, Option 5, Option 6	Ideal

Figure 12. Results view

The “Dashboards” button leads to the Dashboard view.

5.8 Dashboards

Dashboards view has 4 options to present the results:

- Bar

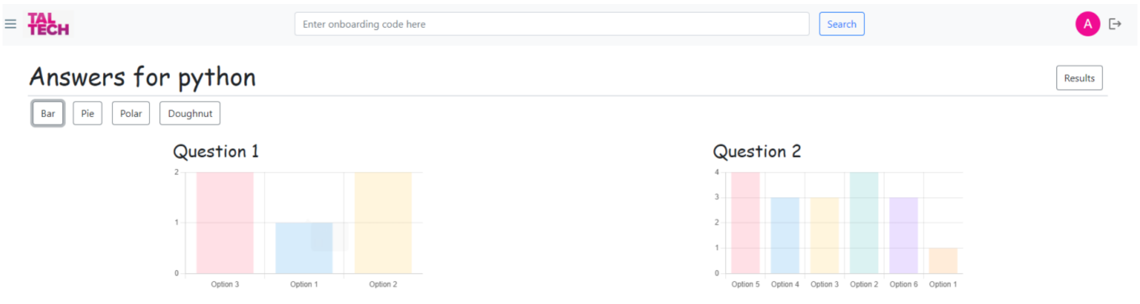


Figure 13. Bar chart

- Pie (When hovering the mouse on the needed part of the graph, numbers of answers will be shown, this works for both pie and doughnut charts)

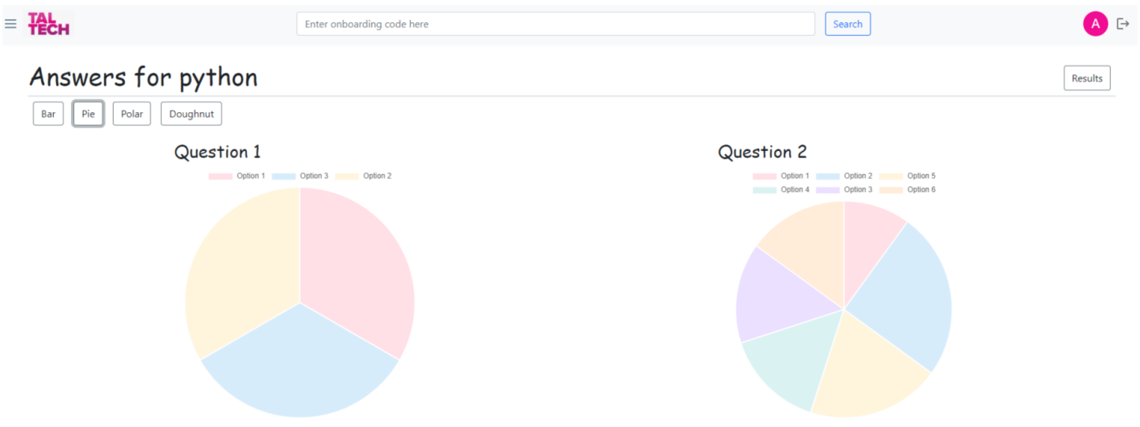


Figure 14. Pie chart

- Polar Area

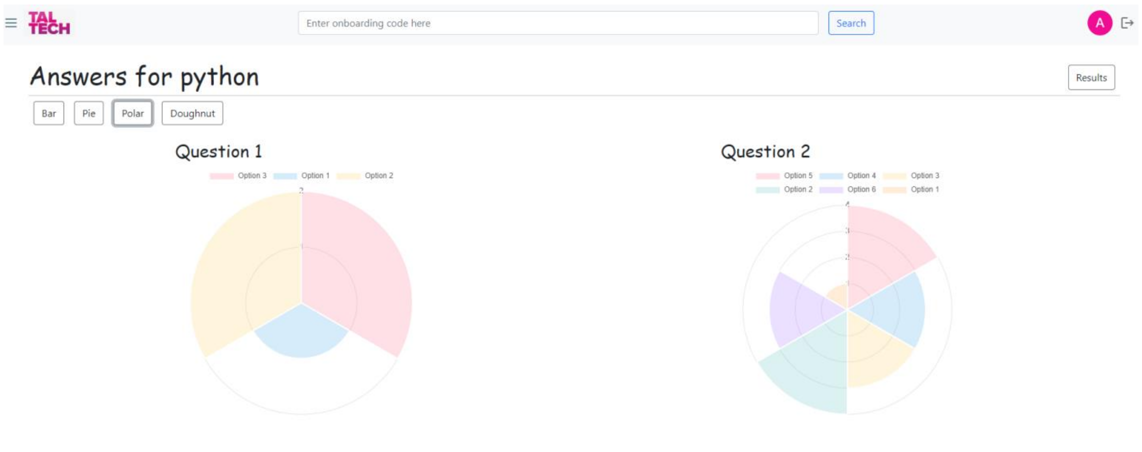


Figure 15. Polar area chart

- Doughnut

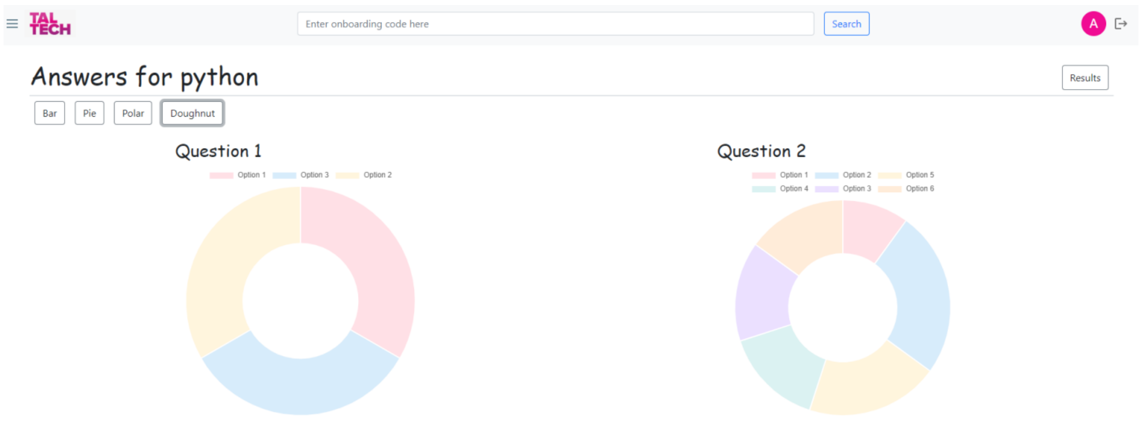


Figure 16. Doughnut chart

6. Discussion

During the development of this project, we had some difficulties, as most of the technologies that we used were new to us. Also, we completed most goals, but not all of them, so future development steps will be discussed here along with difficulties.

6.1 Difficulties

Below are listed some difficulties we had while developing the project.

6.1.1 Project structure

As the project was built from scratch and all we had was just requirements, it was hard to decide what should be the structure of the project and how to realize it. Project structure and the choice of technologies took a while to discuss and make the decisions.

6.1.2 Storing the onboarding

One of the main difficulties at the start of developing the project was saving onboarding data. During analysis, many variants were discussed, such as separate HTML or Markdown files or using the database for all related data saving. Eventually, we decided to store onboarding data in JSON files, because it was the most suitable way to save and restore data.

6.1.3 Containerizing and deploying the project

As we were not familiar enough with Docker, it was hard to containerize the project. All components of the project were successfully put into containers. Deploying the project also took a lot of time due to a lack of experience in deploying the dockerized project. Docker containers are currently running locally and on the server.

6.1.4 Searching for analogs

As far as a project with page generation is not a popular topic, it was complicated to find any good example or helpful information regarding this matter.
First of all, we took a look at a service called WordPress [21], which allows users to create a separate web page with all customizable configurations. This solution was not suitable for the current project because of many factors. Firstly, this service is not free, and users

must choose a plan (monthly or yearly) to start using the service. Secondly, the main goal of the project was to create a service that allows lecturers, who are not familiar with web development, to easily create new onboarding pages and collect feedback. Unluckily, to start using WordPress, a lot of the time and experience are required. Finally, the project has to be the TalTech University's own, and it is inappropriate for our purposes to use such a service as WordPress.

We analyzed the existing solutions, such as Google Forms [22] and Microsoft Forms [23], which were more suitable for the current project. Both services have advantages and disadvantages, which are listed below.

Google Forms advantages:

- Allows adding all needed blocks (text, images, video, surveys).
- It is possible to check all answers and graphs.
- Information pages without surveys are also allowed to be created.
- The service allows changing themes using predefined options (colors, fonts, header image).

Google Forms disadvantages:

- Not possible to add or improve functionality, for example, add additional styling to the form.
- The service does not support Uni-ID authentication.
- Uncomfortable view. The design is too thin, and every block is separated.

Microsoft Forms advantages:

- The service supports Uni-ID because Uni-ID uses Microsoft authentication.
- It is possible to check all answers and graphs.
- The service allows changing the background theme (color or image).

Microsoft Forms disadvantages:

- The service supports adding only survey blocks.
- The service is not supporting adding information pages, as it is not possible to add content (text, image, video) blocks.
- Not possible to add or improve functionality.
- Uncomfortable view. The design is too thin.

Summing up all pros and cons of the existing solutions, the decisions about functionality have been made.

6.2 Future development steps

Below are described the most important future development steps.

6.2.1 Uni-ID authentication & roles

Uni-ID authentication is needed to be implemented in the project. It is needed to grant roles and permissions to the users. At the current stage of development, it is not yet integrated. Currently, a basic login form has been added to restrict unauthorized users from creating, editing, deleting the onboarding, and results viewing. Unauthorized users are only allowed to complete shared onboardings. Authorized users have access to all functionality.

6.2.2 Additional text editing

Also, it is needed to add the rich editor to the text input to add more text editing options such as bulleted lists, different font sizes, highlightings, etc. For now, it is possible to just enter the default text.

6.2.3 Additional style editing

The project is built with predefined styles and is not customizable for the users. For future development, the possibility of editing page styles, colors, and fonts will allow the lecturers to make their onboardings unique.

6.2.4 Templates

To simplify the creation of the onboarding templates can be added. Templates are pre-designed onboardings with the entered initial data to make this process more intuitive for lecturers. Template types must be different so they can be for the first meeting purposes, for the studying purposes during the course (information pages), or some other purposes. Also, additional styling templates can be integrated.

6.2.5 Administration of the study areas

As far as the project is mainly developed for the TalTech University, the feature that has to be developed in the future is study areas administration. Every study area must have a manager, who will be responsible for managing all created onboarding in the particular area.

6.2.6 Mobile version

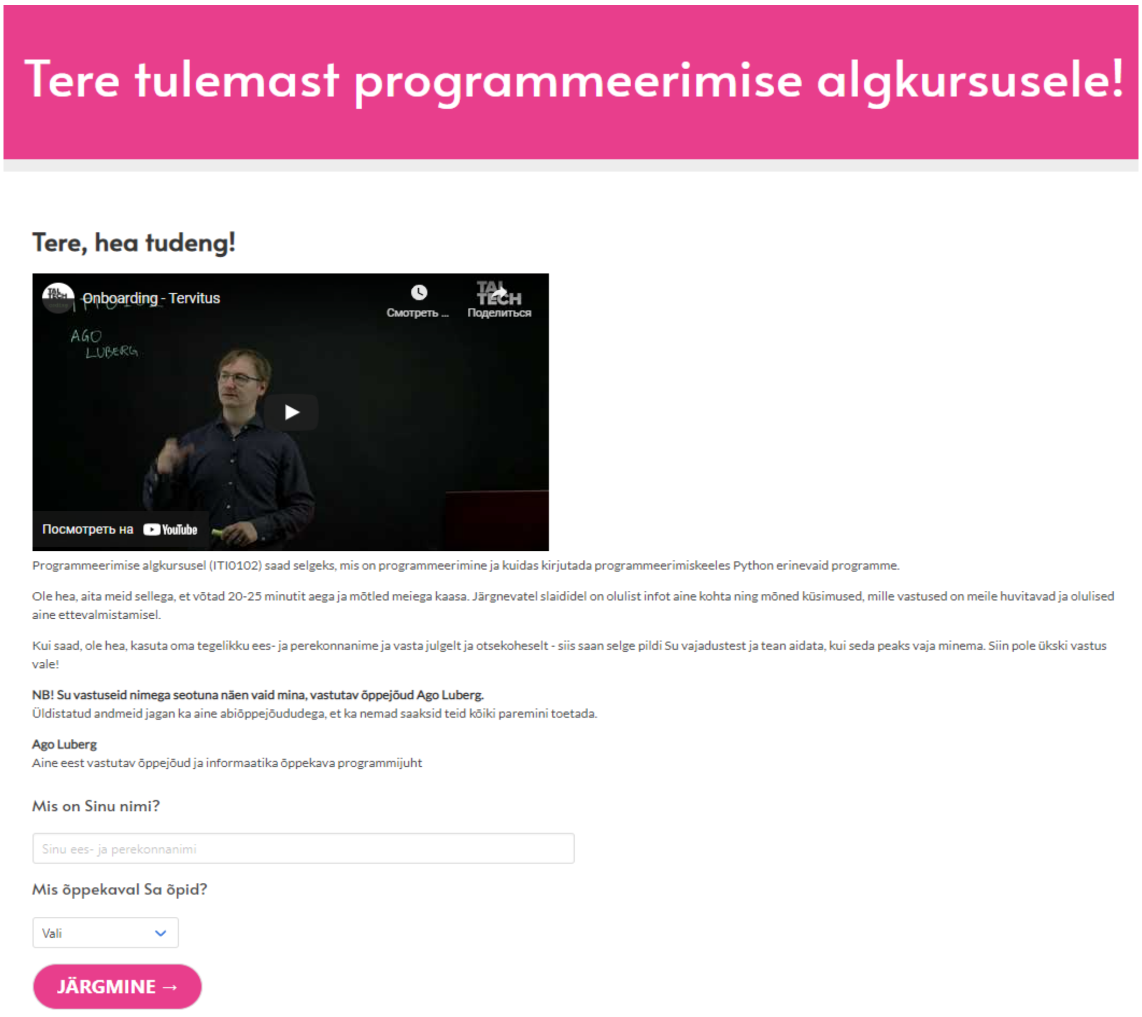
Currently, this service is not supported for mobile phones, but only for computers and tablets. Further developments may be related to optimizing the project and creating a mobile app, which will support onboarding completion for the students. Also, the functionality of the onboarding creation for lecturers could be added to the mobile app.

6.2.7 Third-party use

Allow other universities or companies to use the developed system to create onboardings for their purposes.

7. Summary

The main goal of this project was to create an application that allows creating an onboarding page with editing features. Onboardings can be used to familiarize students with the course, collect answers and feedback, and minimize the dropout rate of the students. In addition, onboarding pages can be used as informative pages in the middle of the course to provide students with comprehensive information, instructions, articles, etc.
The main goal was reached. It’s possible to create onboardings and store data provided by them. Lecturers can add text, pictures, videos, and questionnaires and see all results and the visualized data in the dashboard view. Onboardings can be shared with students using link from share view.
The given example of the view with the desired result that we should reach is shown in Figure 17.



Tere tulemast programmeerimise algkursusele!

Tere, hea tudeng!

Programmeerimise algkursusel (ITI0102) saad selgeks, mis on programmeerimine ja kuidas kirjutada programmeerimiskeeles Python erinevaid programme.

Ole hea, aita meid sellega, et võtad 20-25 minutit aega ja mõtled meiega kaasa. Järgnevatel slaididel on olulist infot aine kohta ning mõned küsimused, mille vastused on meile huvitavad ja olulised aine ettevalmistamisel.

Kui saad, ole hea, kasuta oma tegelikku ees- ja perekonnanime ja vasta julgelt ja otsekoheselt - siis saan selge pildi Su vajadustest ja tean aidata, kui seda peaks vaja minema. Siin pole ükski vastus vale!

NB! Su vastuseid nimega seotuna näen vaid mina, vastutav õppejõud Ago Luberg.
Üldistatud andmeid jagan ka aine abiõppejõududega, et ka nemad saaksid teid kõiki paremini toetada.

Ago Luberg
Aine eest vastutav õppejõud ja informaatika õppekava programmijuht

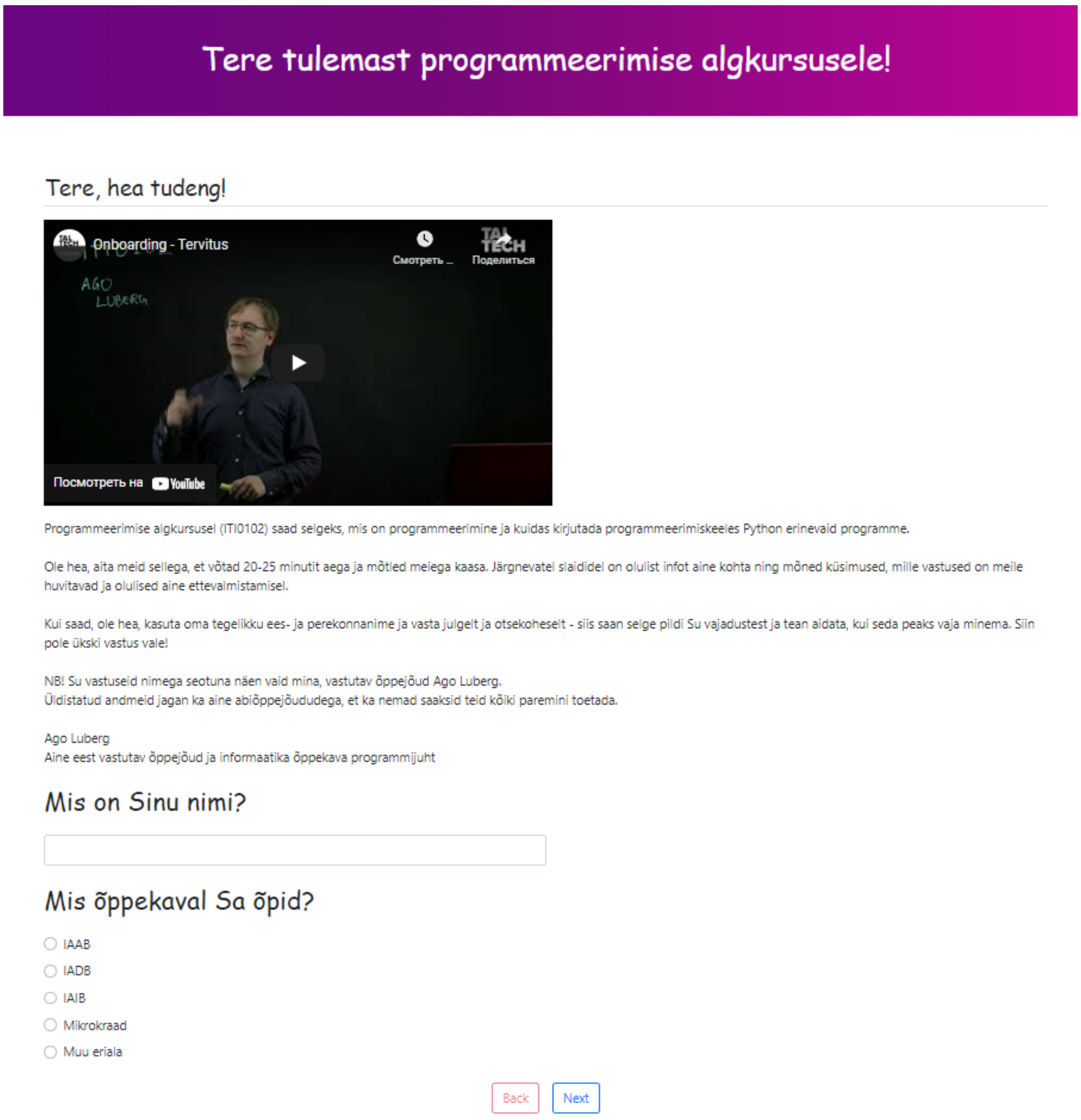
Mis on Sinu nimi?

Sinu ees- ja perekonnanimi

Mis õppekaval Sa õpid?

Figure 17. Desired result

The result which was made using the developed system is shown in Figure 18.


Tere tulemast programmeerimise algkursusele!

Tere, hea tudeng!

Programmeerimise algkursusel (ITI0102) saad selgeks, mis on programmeerimine ja kuidas kirjutada programmeerimiskeeles Python erinevaid programme.

Ole hea, aita meid sellega, et võtad 20-25 minutit aega ja mõtled meiega kaasa. Järgnevatel slaididel on olulist infot aine kohta ning mõned küsimused, mille vastused on meile huvitavad ja olulised aine ettevalmistamisel.

Kui saad, ole hea, kasuta oma tegelikku ees- ja perekonnanime ja vasta julgelt ja otsekoheselt - siis saan selge pildi Su vajadustest ja tean aidata, kui seda peaks vaja minema. Siin pole ükski vastus vale!

NB! Su vastuseid nimega seotuna näen vaid mina, vastutav õppejõud Ago Luberg.
Üldistatud andmeid jagan ka aine abiõppejõududega, et ka nemad saaksid teid kõiki paremini toetada.

Ago Luberg
Aine eest vastutav õppejõud ja informaatika õppekava programmijuht

Mis on Sinu nimi?

Mis õppekaval Sa õpid?

- ○ IAAB
- ○ IADB
- ○ IAIB
- ○ Mikrokraad
- ○ Muu eriala

Figure 18. Generated result

Now it is possible to create such types of onboarding, and the goal may be considered accomplished.

There were a lot of difficulties during the development of this project. All of them are described in the "Discussion" section. Most goals were completed, but not all of them. This project has a huge potential for further use by TalTech University and not only by it. It can be used to introduce curricula and courses, as well as for creating informative pages.

References

[1] “PostgreSQL”, Link: https://www.postgresql.org
[2] “Node.js”, Link: https://nodejs.org/en/
[3] “Representational state transfer”, Link:
https://en.wikipedia.org/wiki/Representational_state_transfer
[4] “React”, Link: https://reactjs.org/
[5] “Single-page application”, Link: https://en.wikipedia.org/wiki/Single-page_application
[6] “Axios”, Link: https://axios-http.com
[7] “Gitlab”, Link: https://about.gitlab.com
[8] “npm”, Link: https://www.npmjs.com/
[9] “ReactPlayer”, Author: Pete Cook, 2022, Link: https://github.com/CookPete/react-player
[10] “Universal cookies”, Author: Benoit Tremblay, 2020, Link:
https://github.com/reactivestack/cookies/tree/master/packages/universal-cookie
[11] “Cookie”, Link: https://en.wikipedia.org/wiki/HTTP_cookie
[12] “react-beautiful-dnd”, Authors: Alex Reardon, Daniel Del Core, 2021, Link:
https://github.com/atlassian/react-beautiful-dnd
[13] “React Icons”, Authors: Goran Gajic, Nolan Leung, 2021, Link: https://github.com/react-icons/react-icons
[14] “react-confirm-alert”, Author: Thawatchai Khuansombat, 2022, Link:
https://github.com/GA-MO/react-confirm-alert
[15] “react-custom-alert”, Author: Hyeon Gyu Kim, 2022, Link:
https://github.com/gusrb3164/react-custom-alert
[16] “React Uuid”, Author: Rick Brown, 2019, Link: https://www.npmjs.com/package/react-uuid
[17] “React Router DOM”, Authors: Michael Jackson, Tim Dorr, Chance Strickland, 2022,
Link: https://github.com/remix-run/react-router/tree/main/packages/react-router-dom
[18] “MUI”, Authors: Olivier Tassinari, Matt (mbrooks), 2022, Link:
https://github.com/mui/material-ui
[19] “Chart.js”, Link: https://www.chartjs.org/
[20] “Bootstrap”, Link: https://en.wikipedia.org/wiki/Bootstrap_(front-end_framework)
[21] “WordPress”, Link: https://wordpress.com/features/
[22] “Google Forms”, Link: https://www.google.com/intl/en-GB/forms/about/
[23] “Microsoft Forms”, Link: https://www.microsoft.com/en-us/microsoft-365/online-surveys-polls-quizzes

Appendix 1 – Non-exclusive licence for reproduction and publication of a graduation thesis[1]

We Anton Antonov, Vladislav Poljakov, German Hanmamedov

1. Grant Tallinn University of Technology free licence (non-exclusive licence) for our thesis "Onboarding System", supervised by Ago Luberg

   1.1. to be reproduced for the purposes of preservation and electronic publication of the graduation thesis, incl. to be entered in the digital collection of the library of Tallinn University of Technology until expiry of the term of copyright;

   1.2. to be published via the web of Tallinn University of Technology, incl. to be entered in the digital collection of the library of Tallinn University of Technology until expiry of the term of copyright.

2. We are aware that the author also retains the rights specified in clause 1 of the non-exclusive licence.

3. We confirm that granting the non-exclusive licence does not infringe other persons' intellectual property rights, the rights arising from the Personal Data Protection Act or rights arising from other legislation.

30.05.2022

[1] The non-exclusive licence is not valid during the validity of access restriction indicated in the student's application for restriction on access to the graduation
thesis that has been signed by the school's dean, except in case of the university's right to reproduce the thesis for preservation purposes only. If a graduation thesis
is based on the joint creative activity of two or more persons and the co-author(s) has/have not granted, by the set deadline, the student defending his/her
graduation thesis consent to reproduce and publish the graduation thesis in compliance with clauses 1.1 and 1.2 of the non-exclusive licence, the non-exclusive
license shall not be valid for the period.