Master Team Project WiSe 2021/2022

Pick-A-Tutor

Milestone 4

GDSD Project Team A

25.03.2022

Team Members

Name	Email ID	Special role
Alexander Wiegel	alexander.wiegel@ai.hs-fulda.de	Team Lead
Rakesh Kumar	rakesh.kumar@ai.hs-fulda.de	Backend Lead
Abdullah Butt	abdullah.butt@ai.hs-fulda.de	
Chahat Soni	chahat.soni@ai.hs-fulda.de	Frontend Lead
Maksim Kanushin	maksim.kanushin@ai.hs-fulda.de	Github Master
Ziad Khaled	ziad.khaled-mohamed@ai.hs-fulda.de	

Revision History

Revision	Date Submitted	Date Revision	Changes

Table of contents

1 Product summary

"Pick-A-Tutor" is a website for students of Fulda University of Applied Sciences to connect with potential tutors who can help them with their studies. It offers following major functionalities:

1. Browse courses or tutors: Anonymous users, students and tutors are able to browse through the available courses or tutors.

2. Search courses or tutors: Anonymous users, students and tutors are able to search the available courses or tutors.

3. Register: Any Fulda university member (whose email suffix string has "hs-fulda.de") can register on the website.

4. Login: Students, tutors and administrators can log in to Pick-A-Tutor based on their respective role.

6. View review: Students and tutors can read the reviews for a course.

9. Send message: Students can contact a tutor or an administrator via on-site messaging. The messages will persist for them to view or reply to.

10. Write review: Students can write reviews and provide ratings for a course. These reviews are shown on the detail page of the respective course.

12. Modify profile: Tutors are allowed to add or update their name, description and pricing.

14. Modify profile image: Tutors can upload and replace a profile picture, which then is to be approved by the admin.

15. Modify and uploads: Tutors can upload files, which then have to be approved by the admin, and delete them.

16. Reply message: Tutors can reply to messages they received.

17. Write message as tutor: Tutors can write new messages to an admin.

19. Write message as admin: Administrators can write new messages to students or tutors or reply to messages.

20. Approval: Administrators can approve any file tutors uploaded in their profile or for a course.

This website offers a **unique feature** of showing the "tutor of the month" based on the average rating provided by students for the course of tutor.

URL: http://20.113.25.17/

2 Usability test plan

Test objectives

The goal of this usability test is to test one feature on the website Pick-A-Tutor, which is a web project done by six master students for the course “Global Distributed Software Development”. Pick-A-Tutor is a tutoring platform exclusive to members of Fulda University of Applied Sciences and brings students and tutors together. Users can register and login, browse and search for the available courses and tutors, filter and sort the results, write reviews, chat with each other and more. The feature which will be looked upon in this test is searching for courses on the “Browse” page. The reason for this being that it is one of the main functions of the website which is very likely to be used by every user or at least the vast majority.

Test background and setup

System setup

The test will be held on a laptop with Windows 11 and the Mozilla Firefox browser (v98.0.2). Apart from the built-in keyboard and a wireless mouse, no other devices will be available for the subject.

Starting point

Since no registration or login is necessary to search for courses, subjects will directly start the usability test on the website’s “Browse” page using the URL below. From there, subjects can search for a specific course and test additional features like filtering and sorting.

Intended users

The subjects should and will have the same origin as the website’s main target group: students from Fulda University of Applied Sciences.

URL

http://20.113.25.17, specifically http://20.113.25.17/browse

What is to be measured

In this usability test we want to measure the user satisfaction for the search feature. We will survey this with a short questionnaire containing three Likert scale questions which will be handed to the subjects after the task below has been performed.

Usability task description

Instructions given to the subject

Find the course “Fun with flags” by the tutor Sheldon Cooper.

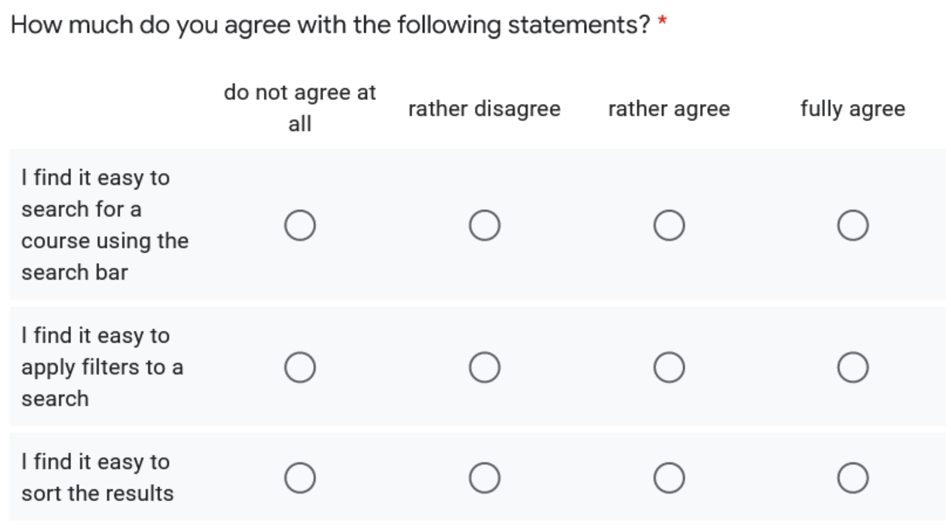
How we would measure **effectiveness**:

After the subjects searched for a course, the results will only consist of courses that contain the sequence of letters they typed into the search bar. One could ensure the search was effective by verifying they can then see the course they were looking for.

How we would measure **efficiency**:

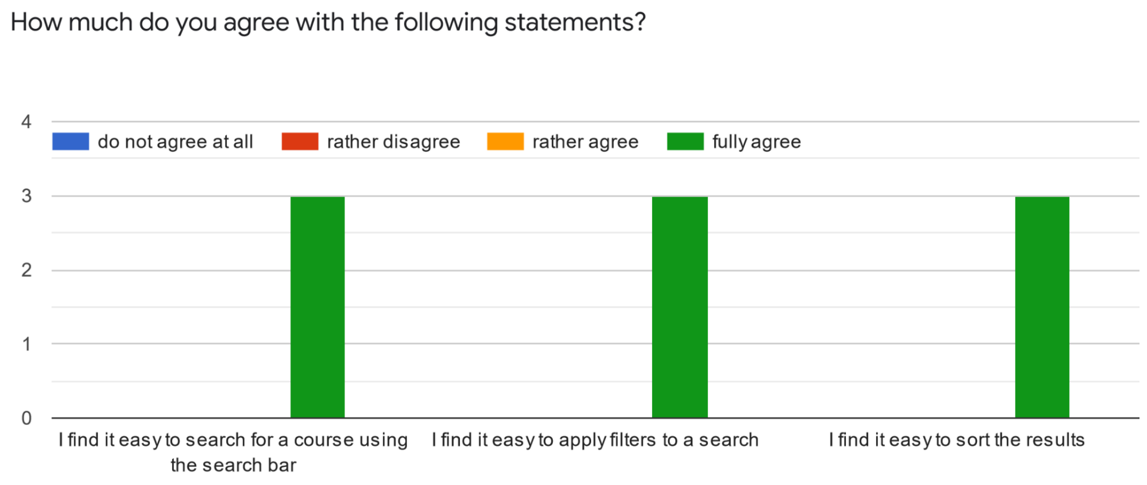
The efficiency of the search could be measured by simply classifying the effort into different levels or even stopping the time it took the subjects to find a specific course that they were looking for.

Likert scale questions to get the user satisfaction



How much do you agree with the following statements? *

	do not agree at all	rather disagree	rather agree	fully agree
I find it easy to search for a course using the search bar	○	○	○	○
I find it easy to apply filters to a search	○	○	○	○
I find it easy to sort the results	○	○	○	○



3 QA test plan

Test objectives

The goal of this QA test is to test one feature on the website Pick-A-Tutor, which is a web project done by six master students for the course “Global Distributed Software Development”. Pick-A-Tutor is a tutoring platform exclusive to members of Fulda University of Applied Sciences and brings students and tutors together. Users can register and login, browse and search for the available courses and tutors, filter and sort the results, write reviews, chat with each other and more. The feature which will be looked upon in this test is searching for courses on the “Browse” page. The reason for this being that it is one of the main functions of the website which is very likely to be used by every user or at least the vast majority.

Hardware and software setup

Tested software:

- Pick-A-Tutor website (milestone4 release) - URL: http://20.113.25.17/browse - running on a remote Azure virtual private server

Hardware used for testing:

- Desktop: Thinkpad T460s running Ubuntu 21.10
- Mobile device: OnePlus 3 running Android 11

Software used for testing:

- Mozilla Firefox browser version 98.0 64 bit for Ubuntu
- Google Chrome browser version 99.0 64 bit for Debian/Ubuntu
- Mozilla Firefox browser version 98.2 64 bit for Android
- Google Chrome browser version 99.0 64 bit for Android
- Mozilla Firefox devtools to mock test different network connectivity conditions

Feature to be tested

Searching for a course and a tutor

QA test plan

#	Title	Description	Input	Expected output	Results
1	View all offered courses	1. Open the browse page and check if the results displayed contain all courses	None	All courses should be displayed	Pass
2	Search for a course on Firefox desktop	1. Enter a course name in the search bar to find relevant courses 2. Enter random input to test the search results are relevant 3. Check if the displayed results are adaptable to mobile and desktop screens 4. Check if the course results are clicked, lead to the corresponding course page	1. Java 2. Organic chemistry 3. lorem ipsum	Courses that are related to the input will be displayed and if no relevant course, only the tutor of the month will be displayed. This behavior should be consistent for anonymous users, logged in tutors and students.	Pass
3	Filter course results based on rating and price	1. Open the browse page and from the filters click on different ratings to make sure the courses are filtered accordingly	1. Click on the different ratings 2. Select different price points 3. Enter min and max values 4. Use rating filters with price filters 5. Select a rating filter and apply it then try to unselect it and check if the results are displayed accordingly	Courses that satisfy the filters should only be displayed. The user should be able to combine the ratings and price filters at the same time. Input validation should be made on the min and max price input areas. User should be to select a filter then deselect it if they wants	Pass but the user has to click on the search button each time to apply the filters. For better UX, the website should be more interactive and display the results instantly when the filters are clicked
4	View all tutors	View all available tutors	1. Click on the course buttons to switch to tutors search	List of all tutors should be displayed	Pass
5	Filter tutors by ratings	1. Open the tutors browse page and from the filters select different ratings	1. Click on different ratings	List of tutors should be displayed according to rating	Pass but the user has to click on the search button each time to apply the filters or the apply filters button For

					better UX, the website should be more interactive and display the results instantly when the filters are clicked
6	Search tutors by name	1. Enter different tutors name to find if they are available on the website	1. Sheldon 2. Alex 3. lorem ipsum	If the tutor is registered, a result corresponding to his page should appear; else no results should be shown	Fail. The results show the page corresponding to the entered tutor name but if the tutor doesn't exist a list of all courses is shown which is not the expected behavior.

4 Code review

by Abdullah Butt to Rakesh Kumar:

```
// File: app\backend\endpoints\customrouter.js
219         const uniqueSuffix =
220             decoded.id +
221             "-" +
222             Date.now() +
223             "-" +
224             Math.round(Math.random() * 1e9);
225         cb(null, "User-" + uniqueSuffix);
```

Abdullah Butt: In Upload Course Files, extension of the uploaded files is not getting saved code in line 224

Rakesh Kumar: Done

```
224  Math.round(Math.random()*1e9)+"."+
file.originalname.split(".").pop();
```

by Maksim Kanushin to Abdullah Butt:

```
// File: app/backend/endpoints/tutorcourses.js
            });
*/
console.log("Tutor Course Updated: " +
existingTutorCourse);
res.json(new SuccessfulResponse("TutorCourse"+
existingTutorCourse.Course.name + " successfully updated",
existingTutorCourse));
```

Maksim Kanushin: I would put [existingTutorCourse] instead of existingTutorCourse into data so the length in the response makes sense

Abdullah Butt: Done

```
// File: app/backend/endpoints/tutorcourses.js
    else {
      return res.json({
        success: false,
        message: "Tutor's Course not found or already deleted",
```

Maksim Kanushin: you can also replace it with res.json(new FailedResponse("Tutor's Course not found or already deleted"))

Abdullah Butt: Done

---

by Rakesh Kumar to Maksim Kanushin:

```
// File: app/backend/endpoints/reviews.js
exports.getReviews = async (req, res) => {
    try {
        let reviews = await Review.findAll({
        include: [
            { model: TutorCourse, include: [Course] },
            { model: User, as: "student" },
            { model: User, as: "tutor" },
        ],
        where: {
            courseId: req.query.courseId,
        },
    });
            res.json(new SuccessfulResponse("Reviews", reviews));
    } catch (e) {
    res.json(new FailedResponse(e.message));
```

```
        }
    };
```

Rakesh Kumar: In all of our code we have created a function and then exported the function at the end of the function. I may keep the code structure the same here as well.

Maksim Kanushin: Done

Rakesh Kumar: Also there is a comment about what function will do in the beginning of each function. I may have a comment in the beginning of each function in this code file.

e.g. `//************* Get All reported reviews ***************`

Maksim Kanushin: Done

---

by Chahat Soni to Alexander Wiegel:

```
// File: app/frontend/src/Components/ApiEndpoints.js
    })
    }

    async function getTutorById(id) {
```

Chahat Soni: Why did you create a new api call here; there is already getListofTutors (wouldn't that work as well) ?

Alexander Wiegel: Sorry I looked if there was already something like "getTutor", but nothing showed up (because it is called "getList..."), so I just wrote my own. I will give it a try, but even if we kept it, it wouldn't do our application any harm apart from adding 7 lines of code. Is "getListOfTutors" even working? It works with a tutor_id that is not even passed as a parameter.

Chahat Soni: You can give it a default value as " "

Alexander Wiegel: Or I use my working endpoint ;) No, seriously now, I could do that later. For now, it's working, so please approve it

---

by Alexander Wiegel to Ziad Khaled:

```
// File: app/frontend/src/Pages/EditTutorProfile.js

0 // you should include a header in the first line marking you as the (main) author of this file

…

// since this is a rather strange syntax, I would suggest to write some inline comments explaining these lines or others may find it confusing

73   return {
74     ...state,
75     [event.name]: event.value,
76   };

…

92 useEffect(() => {
93   getTutorProfile(id)
94 }, []) // you should remove these square brackets, otherwise React will think you forgot to introduce a dependency for your hook

…

// I’m pretty sure you can and should use the strict equality operator === here to really only allow the expected data type

205                 defaultChecked={tutorProfile.gender == 0}
215                 defaultChecked={tutorProfile.gender == 1}
227                 defaultChecked={tutorProfile.gender == 2}
```

---

by Ziad Khaled to Chahat Soni:

```
// File: app/frontend/src/Pages/Home.js

45     <Container style={{ overflowX: 'scroll' }}>
46       <Container style={{ display: 'flex' }}>
```

```
<Container style={{ overflowX: 'scroll' }}>
  <Container style={{ display: 'flex' }}>
<Container style={{ overflowX: 'scroll' }}>
  <Container style={{ display: 'flex' }}>
```

Ziad Khaled: See how you use the same two containers three times in a row to create a horizontal scroll view? Try extracting this into a function to avoid duplicate code.

5 Best practices for security

List major assets you are protecting:

- The tables which provide persistence storage for data are Users (store passwords), User_Profiles, Courses, Tutor_Courses, Course_Additional_Info, Messages, Reviews.
- In addition to that files are being stored on the server at preconfigured paths on the server.

List major threats for each asset above:

- Passwords should not be saved in plain text since can be misused if the database access is compromised.
- Update and delete operations must be protected against unauthorized access.
- Admin routes must be projected against unauthorized access.

For each asset say how you are protecting it:

- Passwords are hashed before being stored in DB.
- The update or delete operations related routes are projected using authentication middleware.
- Administrator routes are also protected through authentication middleware developed for administrators.
- Frontend can only access the files or data stored from the database through APIs which are protected and always require a token for authentication. There is no direct access to db from frontend.

Confirm that you encrypt PW in the DB: We have used the bcryptjs module which encrypts/decrypts the password while storing/retrieving from the database using hash and compare functions. It provides protection against popular attacks such as bruteforce.

Confirmation of input data validation :

1. Login Page :
    a. Requirements :
        i. Email is required
        ii. Password of minimum length of 5 is required
    b. Code : A combination of yup and formik was used
2. Sign Up Page :
    a. Requirements :
        i. Firstname a string is required
        ii. Lastname a string is required

        iii. Email is a email validation (i.e. with @) with suffix allowed ['hs-fulda.de', 'ai.hs-fulda.de', 'informatik.hs-fulda.de']
        iv. Date of birth, to verify a user is at least 18 years of age
        v. Password, at least 5 characters
        vi. Repeat Password, must match Password and be of 5 characters
        vii. Role, either students or tutor; must be defined
        viii. Terms and conditions must be accepted
    b. Code : A combination of yup and formik was used

3. Search bars :
    a. Requirements :
        i. Search field has a maximum input of 40 characters
    b. Code : the input field has a attribute maxLength; which is set to 40
4. Add Course Page :
    a. Requirements :
        i. It should be a new course; conditional validation (if-else) is inplace
        ii. Course name, should not be empty
        iii. Price, should be a number and non empty
    b. Code : A combination of yup and formik was used
5. Edit Course Page :
    a. Requirements :
        i. Course Name, editing not possible
        ii. Price, editable
    b. Code : A combination of yup and formik was used
6. Edit Profile Page :
    a. Requirements :
        i. No null values accepted
    b. Code : A combination of yup and formik was used

6 Adherence to original non-functional specs

Non-functional requirement	Status	Comments
The Application shall be developed, tested and deployed using tools and servers approved by the Class CTO and as agreed in Milestone 0. The Application delivery shall be from the chosen cloud server.	DONE	
The Application shall be optimized for standard desktop/laptop browsers, i.e. must render correctly on the two latest versions of two major browsers.	DONE	
All or selected application functions must render well on mobile devices.	DONE	
Data shall be stored in the database on the team's deployment cloud server.	DONE	
No more than 50 concurrent students shall be accessing the application at any time.	DONE	
The privacy of students shall be protected and all privacy policies will be appropriately communicated to the students.	DONE	
The language used shall be English (no localization needed).	DONE	
The application shall be very easy to use and intuitive.	DONE	
The application should follow established architecture patterns.	DONE	
The application code and its repository shall be easy to inspect and maintain.	DONE	
No email clients shall be allowed.	DONE	
Pay functionality, if any (e.g. paying for goods and services) shall not be implemented nor simulated in UI.	DONE	No pay functionality
Site security: basic best practices shall be applied (as covered in the class) for main data items.	DONE	
The application shall be media rich (images, video, etc.). Media formats shall be standard as used in the market today.	DONE	

Modern SE processes and practices shall be used as specified in the class, including collaborative and continuous SW development.	DONE	
For code development and management, as well as documentation like formal milestones required in the class, each team shall use their own GitHub to be set-up by class instructors and started by each team during Milestone 0.	DONE	
The application UI (WWW and mobile) shall prominently display the following exact text on all pages "Fulda University of Applied Sciences Software Engineering Project, Fall 2021 For Demonstration Only” at the top of the WWW page. (Important so as to not confuse this with a real application).	DONE	