UNIVERSITY OF TRENTO
Department of Information Engineering and Computer Science

Tutor me

Course: “Software Engineering”

DELIVERABLE 2

By:

Filippo Maffei
filippo.maffei@studenti.unitn.it

Massimo Girardelli
massimo.girardelli@studenti.unitn.it

Tetiana Golovkina
tetiana.golovkina@studenti.unitn.it

Trento – 2023

CONTENTS

PROJECT MAIN OBJECTIVE

The main goal of our tutoring system, TUTOR ME, is to create a user-friendly website that connects students and tutors for personalized one-on-one lessons. Our platform makes it easy for students to find the right tutor, schedule lessons, and communicate effectively. Students can browse tutor availability, book lessons, and manage their reservations hassle-free. We provide a dedicated chat feature for seamless communication between students and tutors, where they can ask questions, share files, and access recorded lessons. To help students make informed decisions, we allow them to leave reviews and ratings for tutors they have worked with. Students have the flexibility to adjust their lesson schedules within 24 hours before the lesson. Tutors are promptly compensated after each completed lesson through our secure payment system. Our objective is to create a reliable and efficient platform that simplifies the tutoring experience for both students and tutors.

GIT STRATEGY

We used GitHub to host the code, the branch the main branch is “main”, commits were hosted there or on temporary branches and then merged, https://tutor-me.onrender.com is linked to the latest commit in the main branch and automatically redeployed on every push, the complete git log can be found here: https://github.com/Massiccio1/swe/blob/main/commits.txt and here: https://github.com/Massiccio1/swe/blob/main/git%20log%20oneline.txt

BACKLOG

Volunteers:

- M : Massimo
- F : Filippo
- T : Tanya

Epics:

- Backend
- Frontend

Table 1 – Sprint backlog

ID: backlog Item	sprint task	volunteer	priority	estimated effort	User story points remaining									
					9	8	7	6	5	4	3	2	1	0
0: database reset	database reset functions	M	80	4	2	2	2	2	2	2	2	2	0	0
1: database structure	define database structure and class structure	M,T,F	99	2	2	2	0							
	connection to database	M	98	2	0									
2: deployment	Deploy on render	M	90	4	4	4	2	0						
3: As a user I want to login so that i can browse the site as a logged user	General token	M	89	4	4	3	2	0						
	Student authentication	M		3	3	3	0							
	Tutor authentication	M		3	3	3	0							
4:As a user, I want to see all the available courses	Model relationships with tutors	M,F	40	2	0									
	GET request	M		4	3	0								

Continuation of table 1

ID: backlog Item	sprint task	volunteer	priority	estimated effort	User story points remaining									
					9	8	7	6	5	4	3	2	1	0
5:as a student I want to see a dashboard with all the reservation I made	GET student/me with token	M	50	2	2	2	2	2	0					
6:as a guest, I want to register as a new student	POST student and checks	F	70	2	2	2	2	0						
	check if email is unique	M		1	1	1	1	1	1	1	1	1	0	
7: As a student,i want to reserve a slot	calendar	F	60	2	2	2	2	2	2	2	0			
	POST method	F		2	2	2	2	2	2	2	0			
8: as a tutor, i want to indicate the day in which i will be available	calendar	F	30	2	2	2	2	2	2	2	0			
	date selector	F	30	2	2	2	2	2	2	2	0			
	database	F	20	2	2	2	2	2	2	2	0			
9: as a tutor, I want to have a page to upload teaching materials for students	Add html markup	T	90	2	2	2	2	2	2	2	2	2	0	
	Add js script	T	80	3	3	3	3	3	3	3	3	3	0	
	Add endpoint to get form view	T	40	1	1	1	1	1	1	1	1	1	0	

Continuation of table 1

ID: backlog Item	sprint task	volunteer	priority	estimated effort	User story points remaining									
					9	8	7	6	5	4	3	2	1	0
10: as a tutor, i want to upload teaching materials for the students	new model	T	90	2	2	2	2	2	2	2	2	0		
	new endpoint for upload	T	80	2	2	2	2	2	2	2	2	0		
11: as an admin,i want to ban a user who violates the rules	Add endpoint to ban user	T	20	2	2	2	2	0						
	Extend student model	T	50	1	1	1	1	1	0					
12: as a student i want to delete my reservation	DELETE reservation/id	M	60	2	2	2	2	2	2	2	2	0		
13: as a guest, I want to register as a new student	POST student	F	80	2	2	2	2	2	2	2	2	0		
14: test cases for every macro category	test for student registration	M	10	2	2	2	2	2	2	2	2	2	0	0
	test for student login			2	2	2	2	2	2	2	2	2	0	0
	test for tutor registration			2	2	2	2	2	2	2	2	2	2	2

Continuation of table 1

ID: backlog Item	sprint task	volunteer	priority	estimated effort	User story points remaining									
					9	8	7	6	5	4	3	2	1	0
14: test cases for every macro category	test for tutor login	M	10	2	2	2	2	2	2	2	2	2	0	0
	test for student personal space			2	2	2	2	2	2	2	2	2	2	0
	test for tutor personal space			2	2	2	2	2	2	2	2	2	0	0
	test for prenotation get and get/:id			2	2	2	2	2	2	2	2	2	2	0
	test for new prenotation			2	2	2	2	2	2	2	2	2	0	0
	test for prenotation removal			2	2	2	2	2	2	2	2	2	0	0
	test for course get and get/:id			2	2	2	2	2	2	2	2	2	0	0
	test for course post			2	2	2	2	2	2	2	2	2	0	0
	test for course delete			2	2	2	2	2	2	2	2	2	0	0

REMAINING STORIES IN THE PRODUCT BACKLOG

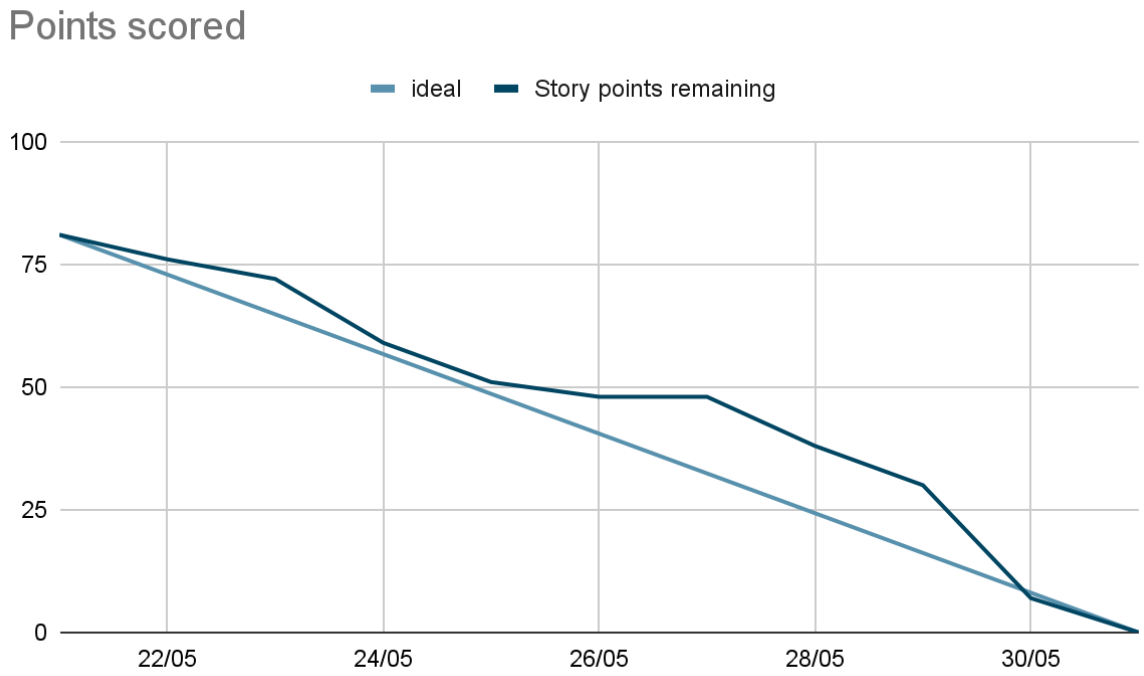
Table 2 – Remaining story 1

user story/item	sprint tasks	priority	volunteer*	estimated effort	real effort
as a tutor i want to start to stream					

Table 3 – Remaining story 2

user story/item	sprint tasks	priority	volunteer*	estimated effort	real effort
as an admin, i want to verify the user data and approve tutors					

BURNDOWN CHART



Picture 1 – Burndown chart with completed work

SPRINT RETROSPECTIVE

What went well:

- Distribution of work and the freedom to choosing ourselves the story to implement, without a pre assigned role from the top;
- Different people with different skills could focus on their area of expertise.

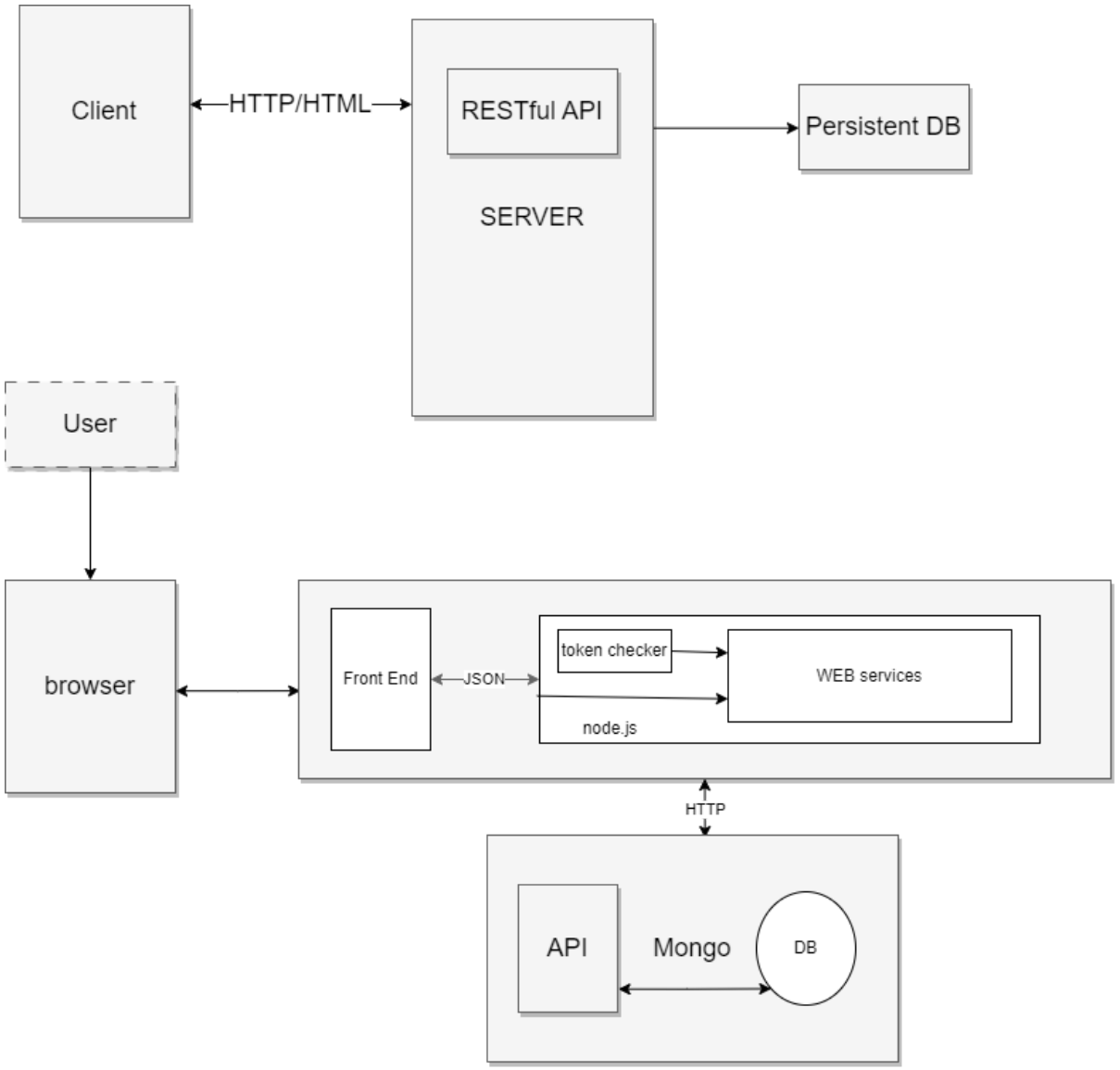
What went badly:

- Time management, too little time everyday to dedicate ourselves to the project to the fullest;
- Some tasks were implemented without following the priority.

What we can improve:

- Better definition of the workflow, some implementation was on hold waiting for other parts to finish;
- Communication in the team;
- Estimation of the task difficulty and the time needed to complete it, many tasks ended up taking a lot more time that estimated;
- The figure of an actual product owner was missing and sometimes the roles were confusing;
- The daily scrum meetings were lacking substantial progress given the little amount of free hours that can be dedicated to the project.

ARCHITECTURE



Picture 2 – Architecture definition

DEMO CREDENTIALS

Table 3 – Demo credentials

Type	Email	Password
Student	e1@gmail.com	p1
	e2@gmail.com	p2
	e3@gmail.com	p3
	e4@gmail.com	p4
Tutor	t1@gmail.com	p1
	t2@gmail.com	p2
	t3@gmail.com	p3
	t4@gmail.com	p4

TESTING

Table 4 – Test cases

Relative url (from https://tutor-me.onrender.com/api/v1)	METHOD	Description of request	Parameters	Expected output	Test case
/authentications	POST	post student credentials for login	body.email body.password	if credential match, generate a token for the user to use	check for empty fields, missing fields, wrong data types
/authentications_tutor	POST	post tutor credentials for login	body.email body.password	if credential match, generate a token for the user to use	check for empty fields, missing fields, wrong data types
/students	GET	get the list of all students	none	list of all students	check if all students are being retrieved
	POST	posts a new student	body.email body.password	generates a new student in the database	check for empty fields, missing fields, wrong data types and if a student already exists with the same email
/student/me	GET	get personal information	query.token	prints information about the student and all the prenotations made	check for empty fields, missing fields, wrong data types, and if the token is for a student or a tutor

Continuation of table 4

Relative url (from https://tutor-me.onrender.com/api/v1)	METHOD	Description of request	Parameters	Expected output	Test case
/student/{id}	GET	get email	none	prints the email related to the student id	check that email and id match the student and no other information in sent
/student/ban	POST	bans student	query.token	student should be flagged as banned in the database	check if the token is a token from an admin
/tutors	GET	get the list of all tutors	none	list of all tutors	check if all tutors are being retrieved
	POST	posts a new tutor	body.email body.password body.name body.desc body.slot	generates a new tutor in the database	check for empty fields, missing fields, wrong data types and if a tutor already exists with the same email
/tutors/me	GET	get personal information	query.token	prints information about the tutor and all the prenotations made and the courses available from that tutor	check for empty fields, missing fields, wrong data types, and if the token is for a student or a tutor, also if all courses and prenotation are displayed
/tutors/{id}	GET	get email	none	prints the email, name, description and slots of a tutor	check if data is correct

Continuation of table 4

Relative url (from https://tutor-me.onrender.com/api/v1)	METHOD	Description of request	Parameters	Expected output	Test case
/prenotations	GET	get all prenotations	query.token	gets all the prenotations related to the student of the token	check if only the prenotations of the correct students are show
	POST	post a new prenotation	query.token body.student body.courseId body.tutor body.timeslot	create a new prenotation in the database	check if all the data match a course and the tutor is available in that slot and if the token matches the studentId
/prenotations/{id}	GET	get info about a single prenotation	query.token	get info about a single prenotation	check if the students or tutor has the rights to access it
	DELETE	deletes a prenotation	query.token	removes prenotation	checks permissions and if the prenotation exists
/course	GET	get all courses	none	returns all courses	check if all courses are being retrieved
/course/{id}	GET	gets a single course	none	return 1 course	check if course exists
/course/new	POST	makes new course	query.token body.tutorId body.desc body.price	makes a new course in the database related to the tutor	check if token in from a tutor and tutorId all fields are present and datatypes

Continuation of table 4

Relative url (from https://tutor-me.onrender.com/api/v1)	METHOD	Description of request	Parameters	Expected output	Test case
/course/delete/{id}	DELETE	deletes course	query.token	deletes a course	check if token in from a tutor and tutorId matches, check if the course exists

These routes:

- /students/me
- /students/ban
- /tutors/me
- /prenotations
- /course/new
- /course/delete

Pass through a token checker that decrypts the token with a secret key, if the token is valid the account is passed down the other routers so that the id, email and account type can be checked.

ADDITIONAL LINKS

1. Site:
   1.1. https://tutor-me.onrender.com
   1.2. (since render doesn't keep the app up 24/7, check the status on https://tutor-me.onrender.com/api/v1/status)
2. Swagger API:
   2.1. https://app.swaggerhub.com/apis/MASSIMOGIRARDELLI/TutorME/1.0.0
3. Repository:
   3.1. https://github.com/Massiccio1/swe