MindMerge — Deliverable e 2

Gabriele Benetti, Gioele Bernardini, Luca Fossa Crescini, Luca Sartore

June 23, 2024

Contents

Abstract

This is the Abstract of the deliverable 2

1 Architecture description

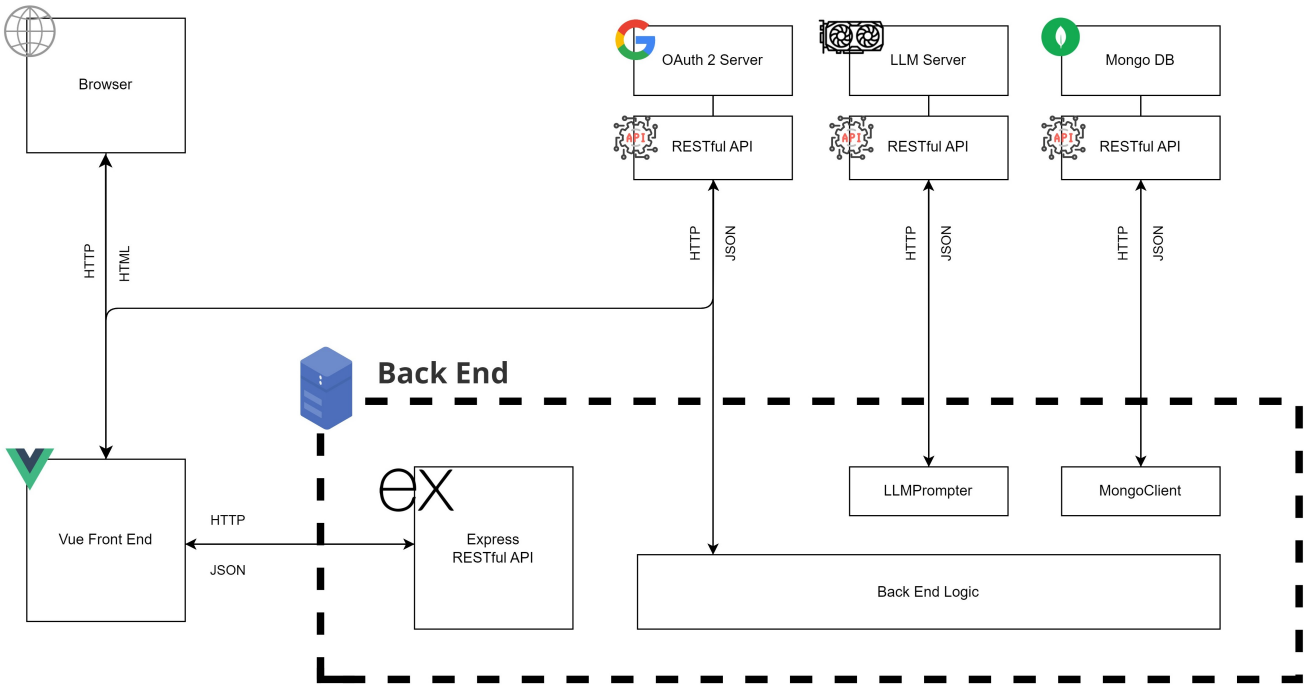


Figure 1: The architecture diagram ot the MindMerge system

The architecture is relatively simple, comprising two classical components: the front end and the back end, along with several smaller components (browser, OAuth server, LLM service, and MongoDB).

The browser communicates with the front end using the HTTP protocol and HTML format.

The front end, based on the Vue library, accesses resources provided by the back end using HTTP with JSON data format.

Authentication is provided by a third-party OAuth server, that communicate with both the front end and back end via HTTP/JSON.

The back end is the most complex component consists of several sub-components, including:

- A part providing RESTful APIs based on Express.js
- A component dedicated to communicating with the database (MongoClient)
- A module for communicating with the LLM service (LLMPrompter)
- Logic containing all functions necessary for the application's operation.

All the communications that the backend has with the external world are made through HTTP with json format.

2 Product Backlog

This section contains the product backlogs we have built during our development process. There are a few differences between the various versions. The most obvious one is that we added a new colum every time to represent the estimated effort that remained for every task. As well as some changes to the descriptions; for example, we realized after we began the first sprint that we missed the "so that.." part of the user stories. Therefore, we added it in the following versions.
The complete list of all the product backlog we submitted, as well as the time they were written, is as follows:

- **Version 1.0:** The initial product backlog, that was written before the sprint 1.
- **Version 2.0:** The second product backlog, tha was written between the sprint 1 and the sprint 2.
- **Version 3.0:** The third product backlog, that was written between the sprint 2 and the sprint 3.
- **Version 4.0:** The fourth product backlog, that was written at the end of the third sprint.

2.1 Version 1.0

Id	*Product Name*	*User story*	*Description*	*Acceptance Criteria*	*How to demo*	*Importance*	*Total Estimated Effort*	**Remaining Estimated Effort (at the beginning of sprint 1)**
1	Account Creation	As a user I want to be able to create an account	A new account can be created	This task is compleated when you can bowse in the mindmerge website, and and create a new account either using google/facebook, or a custom email address. and after that the user is allowed to enter the mind merge platform with his account	Go to the MindMerge website and create a new account using either google/facebook or a custom account. Thes sign in to the website	10	5	5
2	Log In	As a user I want to be able to log in to the service with an account i created	It is possible to log in to the website with an existing account	This task is compleated when an user that is in possession of a MindMerge account is able to successfluy log in	Once you have a mindmerge accoung, go to the sign in page of the mindmerge website and log in	10	5	5
3	Log out	As a user, I want to be able to log out from the app when i no longer nead it	After a successful login, the user is able to log out from the application	This task is compleated when, after clciking the logout button, a log out is performed and the browser is no longer able to access	Go to the Mindmerge website and sign in. After that you should be able to click on the login button, and after that all of the provate pages should be inaccessible	3	2	2
4	Edit account	As a user, I want to be able to edit my account informations	A user should be able to edit his account; Username and passworld in partucular	This task is completed when a user can successfully update their username and password through the account settings page, and the changes are reflected immediately in their account profile	Go to the account settings page of the MindMerge website, update the username and password, save the changes, and verify that the updated information is correctly displayed in the user profile.	1	2	2
5	Delete account	As a user, I want be able to delete my own account at any moment	A user should be able to delete his account permanently	This task is completed when a user can permanently delete their account through the account settings page, and the account is removed from the system without the ability to recover it.	Go to the account settings page of the MindMerge website, select the option to delete the account, confirm the deletion, and verify that the user can no longer log in with that account and all related data is removed.	1	2	2
6	Create organization	As a manager, since I want to use the platform, I want to be able to create an organization at any moment	A user should be able to create an organization, and becoming the owner of it, from the page "Organization view"	This task is completed when a user can successfully create an organization from the "Organization view" page, and the user becomes the owner of the created organization.	Go to the "Organization view" page on the MindMerge website, create a new organization, and verify that the user is listed as the owner of the newly created organization.	9	4	4
7	Add user	As an owner, as my organization grow, I want to be able to add an user at any time	An organization owner should be able to add a user to his organization from the page "Organization View"	This task is completed when an organization owner can successfully add a user to their organization from the "Organization View" page, and the added user is listed as a member of the organization.	Go to the "Organization View" page on the MindMerge website, add a user to the organization, and verify that the new user appears in the organization's member list.	9	3	3
8	Remove user	As an owner, I want to be able to remove a user within my organization at any time	An organization owner should be able to renive a user from his organization from the page "Organization View"	This task is completed when an organization owner can successfully remove a user from their organization from the "Organization View" page, and the removed user is no longer listed as a member of the organization.	Go to the "Organization View" page on the MindMerge website, remove a user from the organization, and verify that the user no longer appears in the organization's member list.	4	2	2
9	License payment	As the owner, I must be able to make the periodic license payments with no issue	The owner of an organization should be able to pay the subscription to the min merge app, using his credit cart from the "Organization View" page	This task is completed when an organization owner can successfully make a periodic license payment using their credit card from the "Organization View" page, and the payment is processed without issues.	Go to the "Organization View" page on the MindMerge website, initiate a license payment, enter credit card details, complete the transaction, and verify that the payment is processed and confirmed.	1	8	8
10	Payment lock	As a user, I shall not be able to use the platform when the periodic payment fails	The app should block specific features after the license to use the MindMerge software is expired	This task is completed when the MindMerge app blocks specific features for a user if the periodic payment fails, indicating that the license to use the software has expired.	The app should prevent access to restricted features and display a notification or message indicating that access is blocked due to an expired license.	1	2	2
11	Create task	As a user, I want to be able to freely create a task at any time	From the home page, an user should be able to create a task/subtask inside an organization	This task is completed when a user can freely create a task at any time from the home page, within an organization, and also create subtasks if needed.	Go to the home page of the MindMerge website, navigate to the desired organization, click on the "Create Task" button, fill in the details for the task, and optionally create subtasks within it.	9	5	5
12	Edit task	As a user, I want to be able to edit the task (e.g. title, description etc)	From the home page, an user should be able to edit a task that he sleected	This task is completed when a user can edit various aspects of a task, such as its title, description, etc., from the home page after selecting the task.	Go to the home page of the MindMerge website, select the desired task, click on the "Edit" button or icon, make the necessary changes to the task details (title, description, etc.), and save the modifications.	8	5	5

13	Delete task	As a user, I want to be able to delete the task at any given time	From the home page, a user should be able to delete a specific task, and all his sub tasks	This task is completed when a user can delete a specific task and all its subtasks from the home page.	Go to the home page of the MindMerge website, locate the desired task, click on the "Delete" button or icon, confirm the deletion, and verify that the task and all its subtasks are permanently removed from the organization's task list.	5	1	1
14	Visualize tasks	As a user, I want to be able to get a visual tree of the tasks and sub-tasks	From the home page, a user should be able to see all the tasks of an organization in an intuitive tree like structure	This task is completed when a user can visualize all the tasks and sub-tasks of an organization in an intuitive tree-like structure from the home page.	Go to the home page of the MindMerge website, navigate to the desired organization, and view the tasks displayed in a hierarchical tree structure where parent tasks are listed along with their respective sub-tasks.	8	5	5
15	Permission system	As the owner, I want a permission system to provided not to permit malicious behaviors	When a user try to see/delete/edit/create a task, the system should ceck if it has the permission before allowing the request	This task is completed when the MindMerge system implements a permission system that checks whether a user has the appropriate permissions before allowing actions such as viewing, deleting, editing, or creating tasks.	Go to the MindMerge website, attempt to perform actions such as viewing, deleting, editing, or creating tasks, and verify that the system checks the user's permissions before allowing the request to proceed.	3	4	4
16	Automatic reports	As a manager, I want automatic reports to be asked and generated to optimize time management	From the home page, a user should be able to ask a question to the system, and the system should be able to generate a answare using LLM tecnoloby and the organization context	This task is completed when a user can ask a question from the home page, and the system, utilizing LLM technology and the context of the organization, generates an answer automatically.	Go to the home page of the MindMerge website, input a question into the system, and verify that the system generates an answer using LLM technology and considers the context of the organization to provide a relevant response.	9	7	7
17	Manual reports	As a manager, I want manual reports to be asked to my sub employes, to gather critical infos	From the home page, a user should be ablle to ask a question, that the system forward the most appropriate user to get an answare.	This task is completed when a user can request a manual report from specific employees by asking a question from the home page, and the system forwards the inquiry to the most appropriate user to provide an answer.	Go to the home page of the MindMerge website, input a question into the system, and verify that the system forwards the inquiry to the appropriate user to provide a manual report.	7	2	2
18	Schedule reports	As a manager, I want to define a recurring request for the reports	From the "Report Page" an user should be able to schedule some reports, such as a specific report answering a specific question is delivered to them every x ammount of time	This task is completed when a manager can define recurring requests for reports from the "Report Page," allowing specific reports to be delivered to them at scheduled intervals.	Go to the "Report Page" on the MindMerge website, select the desired report, specify the frequency (e.g., daily, weekly, monthly), and verify that the system delivers the report answering a specific question to the user at the defined intervals.	8	4	4
19	External notifications	As a user, I want to be able to see major updates at a glance with an external notification system	The system shoud be able to send email notification to the users when neaded	This task is completed when the MindMerge system implements an external notification system to send email notifications to users for major updates.	Go to the MindMerge website, perform actions that trigger major updates, and verify that users receive email notifications informing them of the updates.	8	3	3
20	Internal notifications	As a user, I want to be able to see any update withing the platform as I am working on it	The system shoud be able to send in-app norification to the users, that can be viewed from the "Notification View" page	This task is completed when the MindMerge system implements an internal notification system to send in-app notifications to users for any updates within the platform.	Go to the MindMerge website, perform actions that trigger updates, and verify that users receive in-app notifications that can be viewed from the "Notification View" page.	8	6	6
21	Enable/disable notifications	As a user, I want to be able to turn on/off notifications at my will	An user should be able to enable/disable notification for a specific task from the home page	This task is completed when a user can toggle notifications on or off for specific tasks from the home page.	Go to the home page of the MindMerge website, navigate to the desired task, and verify that users can enable or disable notifications for that task.	5	2	2
22	Testing	As a customer, I want the product to be as relayable and free of bugs	The system shoud have automated and manual tests to verify the correctness of the code.	This task is completed when the MindMerge system has both automated and manual tests in place to verify the correctness of the code, ensuring reliability and minimal bugs.	Run the test command and see the test output	10	10	10
23	Deployment	As a customer, I want to recieve timely and frequent updates that improove functionalityes	The system shoud have a mecanism (scripts/github actions) that allows automatic testing and deployment of the new version fo the application	This task is completed when the MindMerge system implements a mechanism, such as scripts or GitHub Actions, for automatic testing and deployment of new versions of the application.	Trigger the deploiment (via script or commit) and verify that the deployment has heppend	10	4	4
24	Database	As a product owner i want my data to safely stored, and not risk data losses or data breack	The system shoud have a relayable and safe database where to store users data	This task is completed when the MindMerge system utilizes a reliable and secure database to store user data, minimizing the risk of data loss or breaches.	Take a look at the database schema on the MongoDb website	10	5	5

2.2 Version 2.0

Id	*Product Name*	*User story*	*Description*	*Acceptance Criteria*	*How to demo*	*Importance*	*Total Estimated Effort*	Remaining Estimated Effort (at the beginning of sprint 1)	Remaining Estimated Effort (at the beginning of sprint 2)
1	Account Creation	As a user, I want to be able to create an account, so that i am able to access to the mind merge services	It is possible to create a MindMerge account, and after that you are able to perform a log in	This task is compleated when you can bowse in the mindmerge website, and and create a new account either using google/facebook, or a custom email address. and after that the user is allowed to enter the mind merge platform with his account	Go to the MindMerge website and create a new account using either google/facebook or a custom account. Thes sign in to the website	10	5	5	5
2	Log In	As a user I want to be able to log in to the service with an account i created, so that i am able to access to the mind merge services	It is possible to log in to the website with an existing account	This task is compleated when an user that is in possession of a MindMerge account is able to successfluy log in	Once you have a mindmerge accoung, go to the sign in page of the mindmerge website and log in	10	5	5	5
3	Log out	As a user, I want to be able to log out from the app when i no longer nead it, so that my personal informations are not accessible by someone else	After a successful login, the user is able to log out from the application	This task is compleated when, after clciking the logout button, a log out is performed and the browser is no longer able to access	Go to the Mindmerge website and sign in. After that you should be able to click on the login button, and after that all of the provate pages should be inaccessible	3	2	2	2
4	Edit account	As a user, I want to be able to edit my account information, so that i can have more safety and convinience	A user should be able to edit his account; Username and passworld in partucular	This task is completed when a user can successfully update their username and password through the account settings page, and the changes are reflected immediately in their account profile	Go to the account settings page of the MindMerge website, update the username and password, save the changes, and verify that the updated information is correctly displayed in the user profile.	1	2	2	2
5	Delete account	As a user, I want be able to delete my own account at any moment, so that i am able to remove my personal informaion from the mindmerge database whenever i feel like.	A user should be able to delete his account permanently	This task is completed when a user can permanently delete their account through the account settings page, and the account is removed from the system without the ability to recover it.	Go to the account settings page of the MindMerge website, select the option to delete the account, confirm the deletion, and verify that the user can no longer log in with that account and all related data is removed.	1	2	2	2
6	Create organization	As a manager, since I want to use the platform, I want to be able to create an organization at any moment, so that i can start a new project	A user should be able to create an organization, and becoming the owner of it, from the page "Organization view"	This task is completed when a user can successfully create an organization from the "Organization view" page, and the user becomes the owner of the created organization.	Go to the "Organization view" page on the MindMerge website, create a new organization, and verify that the user is listed as the owner of the newly created organization.	9	4	4	2
7	Add user	As an owner, as my organization grows, I want to be able to add a user at any time, so that he can start working for my organization	An organization owner should be able to add a user to his organization from the page "Organization View"	This task is completed when an organization owner can successfully add a user to their organization from the "Organization View" page, and the added user is listed as a member of the organization.	Go to the "Organization View" page on the MindMerge website, add a user to the organization, and verify that the new user appears in the organization's member list.	9	3	3	2
8	Remove user	As an owner, I want to be able to remove a user within my organization at any time, so that he is no longer able to see informations about my organization	An organization owner should be able to renive a user from his organization from the page "Organization View"	This task is completed when an organization owner can successfully remove a user from their organization from the "Organization View" page, and the removed user is no longer listed as a member of the organization.	Go to the "Organization View" page on the MindMerge website, remove a user from the organization, and verify that the user no longer appears in the organization's member list.	4	2	2	2
9	License payment	As the owner, I must be able to make the periodic license payments with no issue, so that i continue to have access to the mindmerge app	The owner of an organization should be able to pay the subscription to the min merge app, using his credit cart from the "Organization View" page	This task is completed when an organization owner can successfully make a periodic license payment using their credit card from the "Organization View" page, and the payment is processed without issues.	Go to the "Organization View" page on the MindMerge website, initiate a license payment, enter credit card details, complete the transaction, and verify that the payment is processed and confirmed.	1	8	8	8
10	Payment lock	As a user, I shall not be able to use the platform when the periodic payment fails, so that I am incentivise to pay	The app should block specific features after the license to use the MindMerge software is expired	This task is completed when the MindMerge app blocks specific features for a user if the periodic payment fails, indicating that the license to use the software has expired.	The app should prevent access to restricted features and display a notification or message indicating that access is blocked due to an expired license.	1	2	2	2
11	Create task	As a user, I want to be able to freely create a task at any time, so that I can annotate the detail of a project	From the home page, an user should be able to create a task/subtask inside an organization	This task is completed when a user can freely create a task at any time from the home page, within an organization, and also create subtasks if needed.	Go to the home page of the MindMerge website, navigate to the desired organization, click on the "Create Task" button, fill in the details for the task, and optionally create subtasks within it.	9	5	5	5
12	Edit task	As a user, I want to be able to edit the task (e.g. title, description etc), so that I can keep the documentation updated.	From the home page, an user should be able to edit a task that he sleected	This task is completed when a user can edit various aspects of a task, such as its title, description, etc., from the home page after selecting the task.	Go to the home page of the MindMerge website, select the desired task, click on the "Edit" button or icon, make the necessary changes to the task details (title, description, etc.), and save the modifications.	8	5	5	5
13	Delete task	As a user, I want to be able to delete the task at any given time, so that I can keep the documentation updated.	From the home page, a user should be able to delete a specific task, and all his sub tasks	This task is completed when a user can delete a specific task and all its subtasks from the home page.	Go to the home page of the MindMerge website, locate the desired task, click on the "Delete" button or icon, confirm the deletion, and verify that the task and all its subtasks are permanently removed from the organization's task list.	5	1	1	5
14	Visualize tasks	As a user, I want to be able to get a visual tree of the tasks and sub-tasks, so that I have a clear idea of what need to be done	From the home page, a user should be able to see all the tasks of an organization in an intuitive tree like structure	This task is completed when a user can visualize all the tasks and sub-tasks of an organization in an intuitive tree-like structure from the home page.	Go to the home page of the MindMerge website, navigate to the desired organization, and view the tasks displayed in a hierarchical tree structure where parent tasks are listed along with their respective sub-tasks.	8	5	5	5
15	Permission system	As the owner, I want a permission system so that I can prevent malicious behaviors	When a user try to see/delete/edit/create a task, the system should ceck if it has the permission before allowing the request	This task is completed when the MindMerge system implements a permission system that checks whether a user has the appropriate permissions before allowing actions such as viewing, deleting, editing, or creating tasks.	Go to the MindMerge website, attempt to perform actions such as viewing, deleting, editing, or creating tasks, and verify that the system checks the user's permissions before allowing the request to proceed.	3	4	4	4

Id	*Product Name*	*User story*	*Description*	*Acceptance Criteria*	*How to demo*	*Importance*	*Total Estimated Effort*	**Remaining Estimated Effort (at the beginning of sprint 1)**	**Remaining Estimated Effort (at the beginning of sprint 2)**
16	Automatic reports	As a manager, I want automatic reports to be asked and generated so that I can easly and quickly know what is going on	From the home page, a user should be able to ask a question to the system, and the system should be able to generate a answare using LLM tecnoloby and the organization context	This task is completed when a user can ask a question from the home page, and the system, utilizing LLM technology and the context of the organization, generates an answer automatically.	Go to the home page of the MindMerge website, input a question into the system, and verify that the system generates an answer using LLM technology and considers the context of the organization to provide a relevant response.	9	7	7	7
17	Manual reports	As a manager, I want manual reports to be asked to my sub employes, so that I can know for certain what is going on	From the home page, a user should be able to ask a question, that the system forward the most appropriate user to get an answare.	This task is completed when a user can request a manual report from specific employees by asking a question from the home page, and the system forwards the inquiry to the most appropriate user to provide an answer.	Go to the home page of the MindMerge website, input a question into the system, and verify that the system forwards the inquiry to the appropriate user to provide a manual report.	7	2	2	2
18	Schedule reports	As a manager, I want to define a recurring request for the reports, so that I will be updated regularly on some tasks	From the "Report Page" an user should be able to schedule some reports, such as a specific report answering a specific question is delivered to them every x ammount of time	This task is completed when a manager can define recurring requests for reports from the "Report Page," allowing specific reports to be delivered to them at scheduled intervals.	Go to the "Report Page" on the MindMerge website, select the desired report, specify the frequency (e.g., daily, weekly, monthly), and verify that the system delivers the report answering a specific question to the user at the defined intervals.	8	4	4	4
19	External notifications	As a user, I want to be able to see major updates at a glance with an external notification system (email), so that I get notified when something important happens	The system shoud be able to send email notification to the users when neaded	This task is completed when the MindMerge system implements an external notification system to send email notifications to users for major updates.	Go to the MindMerge website, perform actions that trigger major updates, and verify that users receive email notifications informing them of the updates.	8	3	3	3
20	Internal notifications	As a user, I want to be able to see any update withing the platform as I am working on it, so that I get notified when something important happens	The system shoud be able to send in-app norification to the users, that can be viewed from the "Notification View" page	This task is completed when the MindMerge system implements an internal notification system to send in-app notifications to users for any updates within the platform.	Go to the MindMerge website, perform actions that trigger updates, and verify that users receive in-app notifications that can be viewed from the "Notification View" page.	8	6	6	6
21	Enable/disable notifications	As a user, I want to be able to turn on/off notifications at my will, so that I don't get overwhelmed by useless notifications	An user should be able to enable/disable notification for a specific task from the home page	This task is completed when a user can toggle notifications on or off for specific tasks from the home page.	Go to the home page of the MindMerge website, navigate to the desired task, and verify that users can enable or disable notifications for that task.	5	2	2	2
22	Testing	As a customer, I want the product to be as relayable and free of bugs as possible, so that it dose not impact my project	The system shoud have automated and manual tests to verify the correctness of the code.	This task is completed when the MindMerge system has both automated and manual tests in place to verify the correctness of the code, ensuring reliability and minimal bugs.	Run the test command and see the test output	10	10	10	5
23	Deployment	As a customer, I want to receive timely and frequent updates that improve functionalities, so that i can always have the latest features	The system shoud have a mecanism (scripts/github actions) that allows automatic testing and deployment of the new version fo the application	This task is completed when the MindMerge system implements a mechanism, such as scripts or GitHub Actions, for automatic testing and deployment of new versions of the application.	Trigger the deploiment (via script or commit) and verify that the deployment has heppend	10	4	4	0
24	Database	As a product owner i want my data to be safely stored, and not to risk data losses or data break, so that my project dose not risk to fail because of data losses	The system shoud have a relayable and safe database where to store users data	This task is completed when the MindMerge system utilizes a reliable and secure database to store user data, minimizing the risk of data loss or breaches.	Take a look at the database schema on the MongoDb website	10	5	5	0

2.3 Version 3.0

Id	*Product Name*	*User story*	*Description*	*Acceptance Criteria*	*How to demo*	*Importance*	*Total Estimated Effort*	**Remaining Estimated Effort (at the beginning of sprint 1)**	**Remaining Estimated Effort (at the beginning of sprint 2)**	**Remaining Estimated Effort (at the beginning of sprint 3)**
1	Account Creation	As a user, I want to be able to create an account, so that i am able to access to the mind merge services	It is possible to create a MindMerge account, and after that you are able to perform a log in	This task is compleated when you can bowse in the mindmerge website, and and create a new account either using google/facebook, or a custom email address. and after that the user is allowed to enter the mind merge platform with his account	Go to the MindMerge website and create a new account using either google/facebook or a custom account. Thes sign in to the website	10	5	5	5	0
2	Log In	As a user I want to be able to log in to the service with an account i created, so that i am able to access to the mind merge services	It is possible to log in to the website with an existing account	This task is compleated when an user that is in possession of a MindMerge account is able to successfluy log in	Once you have a mindmerge accoung, go to the sign in page of the mindmerge website and log in	10	5	5	5	0
3	Log out	As a user, I want to be able to log out from the app when i no longer nead it, so that my personal informations are not accessible by someone else	After a successful login, the user is able to log out from the application	This task is compleated when, after clciking the logout button, a log out is performed and the browser is no longer able to access	Go to the Mindmerge website and sign in. After that you should be able to click on the login button, and after that all of the provate pages should be inaccessible	3	2	2	2	0
4	Edit account	As a user, I want to be able to edit my account information, so that i can have more safety and convinience	A user should be able to edit his account; Username and passworld in partucular	This task is completed when a user can successfully update their username and password through the account settings page, and the changes are reflected immediately in their account profile	Go to the account settings page of the MindMerge website, update the username and password, save the changes, and verify that the updated information is correctly displayed in the user profile.	1	2	2	2	2
5	Delete account	As a user, I want be able to delete my own account at any moment, so that i am able to remove my personal informaion from the mindmerge database whenever i feel like.	A user should be able to delete his account permanently	This task is completed when a user can permanently delete their account through the account settings page, and the account is removed from the system without the ability to recover it.	Go to the account settings page of the MindMerge website, select the option to delete the account, confirm the deletion, and verify that the user can no longer log in with that account and all related data is removed.	1	2	2	2	2
6	Create organization	As a manager, since I want to use the platform, I want to be able to create an organization at any moment, so that i can start a new project	A user should be able to create an organization, and becoming the owner of it, from the page "Organization view"	This task is completed when a user can successfully create an organization from the "Organization view" page, and the user becomes the owner of the created organization.	Go to the "Organization view" page on the MindMerge website, create a new organization, and verify that the user is listed as the owner of the newly created organization.	9	4	4	2	0
7	Add user	As an owner, as my organization grows, I want to be able to add a user at any time, so that he can start working for my organization	An organization owner should be able to add a user to his organization from the page "Organization View"	This task is completed when an organization owner can successfully add a user to their organization from the "Organization View" page, and the added user is listed as a member of the organization.	Go to the "Organization View" page on the MindMerge website, add a user to the organization, and verify that the new user appears in the organization's member list.	9	3	3	2	0
8	Remove user	As an owner, I want to be able to remove a user within my organization at any time, so that he is no longer able to see informations about my organization	An organization owner should be able to renive a user from his organization from the page "Organization View"	This task is completed when an organization owner can successfully remove a user from their organization from the "Organization View" page, and the removed user is no longer listed as a member of the organization.	Go to the "Organization View" page on the MindMerge website, remove a user from the organization, and verify that the user no longer appears in the organization's member list.	4	2	2	2	0
9	License payment	As the owner, I must be able to make the periodic license payments with no issue, so that i continue to have access to the mindmerge app	The owner of an organization should be able to pay the subscription to the min merge app, using his credit cart from the "Organization View" page	This task is completed when an organization owner can successfully make a periodic license payment using their credit card from the "Organization View" page, and the payment is processed without issues.	Go to the "Organization View" page on the MindMerge website, initiate a license payment, enter credit card details, complete the transaction, and verify that the payment is processed and confirmed.	1	8	8	8	8
10	Payment lock	As a user, I shall not be able to use the platform when the periodic payment fails, so that I am incentivise to pay	The app should block specific features after the license to use the MindMerge software is expired	This task is completed when the MindMerge app blocks specific features for a user if the periodic payment fails, indicating that the license to use the software has expired.	The app should prevent access to restricted features and display a notification or message indicating that access is blocked due to an expired license.	1	2	2	2	2
11	Create task	As a user, I want to be able to freely create a task at any time, so that I can annotate the detail of a project	From the home page, an user should be able to create a task/subtask inside an organization	This task is completed when a user can freely create a task at any time from the home page, within an organization, and also create subtasks if needed.	Go to the home page of the MindMerge website, navigate to the desired organization, click on the "Create Task" button, fill in the details for the task, and optionally create subtasks within it.	9	5	5	5	5
12	Edit task	As a user, I want to be able to edit the task (e.g. title, description etc), so that I can keep the documentation updated.	From the home page, an user should be able to edit a task that he sleected	This task is completed when a user can edit various aspects of a task, such as its title, description, etc., from the home page after selecting the task.	Go to the home page of the MindMerge website, select the desired task, click on the "Edit" button or icon, make the necessary changes to the task details (title, description, etc.), and save the modifications.	8	5	5	5	5
13	Delete task	As a user, I want to be able to delete the task at any given time, so that I can keep the documentation updated.	From the home page, a user should be able to delete a specific task, and all his sub tasks	This task is completed when a user can delete a specific task and all its subtasks from the home page.	Go to the home page of the MindMerge website, locate the desired task, click on the "Delete" button or icon, confirm the deletion, and verify that the task and all its subtasks are permanently removed from the organization's task list.	5	1	1	5	5
14	Visualize tasks	As a user, I want to be able to get a visual tree of the tasks and sub-tasks, so that I have a clear idea of what need to be done	From the home page, a user should be able to see all the tasks of an organization in an intuitive tree like structure	This task is completed when a user can visualize all the tasks and sub-tasks of an organization in an intuitive tree-like structure from the home page.	Go to the home page of the MindMerge website, navigate to the desired organization, and view the tasks displayed in a hierarchical tree structure where parent tasks are listed along with their respective sub-tasks.	8	5	5	5	5
15	Permission system	As the owner, I want a permission system so that I can prevent malicious behaviors	When a user try to see/delete/edit/create a task, the system should ceck if it has the permission before allowing the request	This task is completed when the MindMerge system implements a permission system that checks whether a user has the appropriate permissions before allowing actions such as viewing, deleting, editing, or creating tasks.	Go to the MindMerge website, attempt to perform actions such as viewing, deleting, editing, or creating tasks, and verify that the system checks the user's permissions before allowing the request to proceed.	3	4	4	4	4
16	Automatic reports	As a manager, I want automatic reports to be asked and generated so that I can easly and quickly know what is going on	From the home page, a user should be able to ask a question to the system, and the system should be able to generate a answare using LLM tecnoloby and the organization context	This task is completed when a user can ask a question from the home page, and the system, utilizing LLM technology and the context of the organization, generates an answer automatically.	Go to the home page of the MindMerge website, input a question into the system, and verify that the system generates an answer using LLM technology and considers the context of the organization to provide a relevant response.	9	7	7	7	7

Id	*Product Name*	*User story*	*Description*	*Acceptance Criteria*	*How to demo*	*Importance*	*Total Estimated Effort*	Remaining Estimated Effort (at the beginning of sprint 1)	Remaining Estimated Effort (at the beginning of sprint 2)	Remaining Estimated Effort (at the beginning of sprint 3)
17	Manual reports	As a manager, I want manual reports to be asked to my sub employes, so that I can know for certain what is going on	From the home page, a user should be ablle to ask a question, that the system forward the most appropriate user to get an answare.	This task is completed when a user can request a manual report from specific employees by asking a question from the home page, and the system forwards the inquiry to the most appropriate user to provide an answer.	Go to the home page of the MindMerge website, input a question into the system, and verify that the system forwards the inquiry to the appropriate user to provide a manual report.	7	2	2	2	2
18	Schedule reports	As a manager, I want to define a recurring request for the reports, so that I will be updated regularly on some tasks	From the "Report Page" an user should be able to schedule some reports, such as a specific report answering a specific question is delivered to them every x ammount of time	This task is completed when a manager can define recurring requests for reports from the "Report Page," allowing specific reports to be delivered to them at scheduled intervals.	Go to the "Report Page" on the MindMerge website, select the desired report, specify the frequency (e.g., daily, weekly, monthly), and verify that the system delivers the report answering a specific question to the user at the defined intervals.	8	4	4	4	4
19	External notifications	As a user, I want to be able to see major updates at a glance with an external notification system (email), so that I get notified when something important happens	The system shoud be able to send email notification to the users when neaded	This task is completed when the MindMerge system implements an external notification system to send email notifications to users for major updates.	Go to the MindMerge website, perform actions that trigger major updates, and verify that users receive email notifications informing them of the updates.	8	3	3	3	0
20	Internal notifications	As a user, I want to be able to see any update withing the platform as I am working on it, so that I get notified when something important happens	The system shoud be able to send in-app norification to the users, that can be viewed from the "Notification View" page	This task is completed when the MindMerge system implements an internal notification system to send in-app notifications to users for any updates within the platform.	Go to the MindMerge website, perform actions that trigger updates, and verify that users receive in-app notifications that can be viewed from the "Notification View" page.	8	6	6	6	6
21	Enable/disable notifications	As a user, I want to be able to turn on/off notifications at my will, so that I don't get overwhelmed by useless notifications	An user should be able to enable/disable notification for a specific task from the home page	This task is completed when a user can toggle notifications on or off for specific tasks from the home page.	Go to the home page of the MindMerge website, navigate to the desired task, and verify that users can enable or disable notifications for that task.	5	2	2	2	2
22	Testing	As a customer, I want the product to be as relayable and free of bugs as possible, so that it dose not impact my project	The system shoud have automated and manual tests to verify the correctness of the code.	This task is completed when the MindMerge system has both automated and manual tests in place to verify the correctness of the code, ensuring reliability and minimal bugs.	Run the test command and see the test output	10	10	10	5	4
23	Deployment	As a customer, I want to receive timely and frequent updates that improve functionalities, so that i can always have the latest features	The system shoud have a mecanism (scripts/github actions) that allows automatic testing and deployment of the new version fo the application	This task is completed when the MindMerge system implements a mechanism, such as scripts or GitHub Actions, for automatic testing and deployment of new versions of the application.	Trigger the deploiment (via script or commit) and verify that the deployment has heppend	10	4	4	0	0
24	Database	As a product owner i want my data to be safely stored, and not to risk data losses or data break, so that my project dose not risk to fail because of data losses	The system shoud have a relayable and safe database where to store users data	This task is completed when the MindMerge system utilizes a reliable and secure database to store user data, minimizing the risk of data loss or breaches.	Take a look at the database schema on the MongoDb website	10	5	5	0	0

2.4 Version 4.0

Id	*Product Name*	*User story*	*Description*	*Acceptance Criteria*	*How to demo*	*Importance*	*Total Estimated Effort*	Remaining Estimated Effort (at the beginning of sprint 1)	Remaining Estimated Effort (at the beginning of sprint 2)	Remaining Estimated Effort (at the beginning of sprint 3)	Remaining Estimated Effort (at the end of sprint 3)
1	Account Creation	As a user, I want to be able to create an account, so that i am able to access to the mind merge services	It is possible to create a MindMerge account, and after that you are able to perform a log in	This task is compleated when you can bowse in the mindmerge website, and and create a new account either using google/facebook, or a custom email address. and after that the user is allowed to enter the mind merge platform with his account	Go to the MindMerge website and create a new account using either google/facebook or a custom account. Thes sign in to the website	10	5	5	5	0	0
2	Log In	As a user I want to be able to log in to the service with an account i created, so that i am able to access to the mind merge services	It is possible to log in to the website with an existing account	This task is compleated when an user that is in possession of a MindMerge account is able to successfluy log in	Once you have a mindmerge accoung, go to the sign in page of the mindmerge website and log in	10	5	5	5	0	0
3	Log out	As a user, I want to be able to log out from the app when i no longer nead it, so that my personal informations are not accessible by someone else	After a successful login, the user is able to log out from the application	This task is compleated when, after clciking the logout button, a log out is performed and the browser is no longer able to access	Go to the Mindmerge website and sign in. After that you should be able to click on the login button, and after that all of the provate pages should be inaccessible	3	2	2	2	0	0
4	Edit account	As a user, I want to be able to edit my account information, so that i can have more safety and convinience	A user should be able to edit his account; Username and passworld in partucular	This task is completed when a user can successfully update their username and password through the account settings page, and the changes are reflected immediately in their account profile	Go to the account settings page of the MindMerge website, update the username and password, save the changes, and verify that the updated information is correctly displayed in the user profile.	1	2	2	2	2	2
5	Delete account	As a user, I want be able to delete my own account at any moment, so that i am able to remove my personal informaion from the mindmerge database whenever i feel like.	A user should be able to delete his account permanently	This task is completed when a user can permanently delete their account through the account settings page, and the account is removed from the system without the ability to recover it.	Go to the account settings page of the MindMerge website, select the option to delete the account, confirm the deletion, and verify that the user can no longer log in with that account and all related data is removed.	1	2	2	2	2	2
6	Create organization	As a manager, since I want to use the platform, I want to be able to create an organization at any moment, so that i can start a new project	A user should be able to create an organization, and becoming the owner of it, from the page "Organization view"	This task is completed when a user can successfully create an organization from the "Organization view" page, and the user becomes the owner of the created organization.	Go to the "Organization view" page on the MindMerge website, create a new organization, and verify that the user is listed as the owner of the newly created organization.	9	4	4	2	0	0
7	Add user	As an owner, as my organization grows, I want to be able to add a user at any time, so that he can start working for my organization	An organization owner should be able to add a user to his organization from the page "Organization View"	This task is completed when an organization owner can successfully add a user to their organization from the "Organization View" page, and the added user is listed as a member of the organization.	Go to the "Organization View" page on the MindMerge website, add a user to the organization, and verify that the new user appears in the organization's member list.	9	3	3	2	0	0
8	Remove user	As an owner, I want to be able to remove a user within my organization at any time, so that he is no longer able to see informations about my organization	An organization owner should be able to renive a user from his organization from the page "Organization View"	This task is completed when an organization owner can successfully remove a user from the "Organization View" page, and the removed user is no longer listed as a member of the organization.	Go to the "Organization View" page on the MindMerge website, remove a user from the organization, and verify that the user no longer appears in the organization's member list.	4	2	2	2	0	0
9	License payment	As the owner, I must be able to make the periodic license payments with no issue, so that i continue to have access to the mindmerge app	The owner of an organization should be able to pay the subscription to the min merge app, using his credit cart from the "Organization View" page	This task is completed when an organization owner can successfully make a periodic license payment using their credit card from the "Organization View" page, and the payment is processed without issues.	Go to the "Organization View" page on the MindMerge website, initiate a license payment, enter credit card details, complete the transaction, and verify that the payment is processed and confirmed.	1	8	8	8	8	8
10	Payment lock	As a user, I shall not be able to use the platform when the periodic payment fails, so that I am incentivise to pay	The app should block specific features after the license to use the MindMerge software is expired	This task is completed when the MindMerge app blocks specific features for a user if the periodic payment fails, indicating that the license to use the software has expired.	The app should prevent access to restricted features and display a notification or message indicating that access is blocked due to an expired license.	1	2	2	2	2	2
11	Create task	As a user, I want to be able to freely create a task at any time, so that I can annotate the detail of a project	From the home page, an user should be able to create a task/subtask inside an organization	This task is completed when a user can freely create a task at any time from the home page, within an organization, and also create subtasks if needed.	Go to the home page of the MindMerge website, navigate to the desired organization, click on the "Create Task" button, fill in the details for the task, and optionally create subtasks within it.	9	5	5	5	5	0
12	Edit task	As a user, I want to be able to edit the task (e.g. title, description etc), so that I can keep the documentation updated.	From the home page, an user should be able to edit a task that he sleected	This task is completed when a user can edit various aspects of a task, such as its title, description, etc., from the home page after selecting the task.	Go to the home page of the MindMerge website, select the desired task, click on the "Edit" button or icon, make the necessary changes to the task details (title, description, etc.), and save the modifications.	8	5	5	5	5	1
13	Delete task	As a user, I want to be able to delete the task at any given time, so that I can keep the documentation updated.	From the home page, a user should be able to delete a specific task, and all his sub tasks	This task is completed when a user can delete a specific task and all its subtasks from the home page.	Go to the home page of the MindMerge website, locate the desired task, click on the "Delete" button or icon, confirm the deletion, and verify that the task and all its subtasks are permanently removed from the organization's task list.	5	1	1	5	5	0
14	Visualize tasks	As a user, I want to be able to get a visual tree of the tasks and sub-tasks, so that I have a clear idea of what need to be done	From the home page, a user should be able to see all the tasks of an organization in an intuitive tree like structure	This task is completed when a user can visualize all the tasks and sub-tasks of an organization in an intuitive tree-like structure from the home page.	Go to the home page of the MindMerge website, navigate to the desired organization, and view the tasks displayed in a hierarchical tree structure where parent tasks are listed along with their respective sub-tasks.	8	5	5	5	5	1
15	Permission system	As the owner, I want a permission system so that I can prevent malicious behaviors	When a user try to see/delete/edit/create a task, the system should ceck if it has the permission before allowing the request	This task is completed when the MindMerge system implements a permission system that checks whether a user has the appropriate permissions before allowing actions such as viewing, deleting, editing, or creating tasks.	Go to the MindMerge website, attempt to perform actions such as viewing, deleting, editing, or creating tasks, and verify that the system checks the user's permissions before allowing the request to proceed.	3	4	4	4	4	4
16	Automatic reports	As a manager, I want automatic reports to be asked and generated so that I can easly and quickly know what is going on	From the home page, a user should be able to ask a question to the system, and the system should be able to generate a answare using LLM tecnoloby and the organization context	This task is completed when a user can ask a question from the home page, and the system, utilizing LLM technology and the context of the organization, generates an answer automatically.	Go to the home page of the MindMerge website, input a question into the system, and verify that the system generates an answer using LLM technology and considers the context of the organization to provide a relevant response.	9	7	7	7	7	7
17	Manual reports	As a manager, I want manual reports to be asked to my sub employes, so that I can know for certain what is going on	From the home page, a user should be ablle to ask a question, that the system forward the most appropriate user to get an answare.	This task is completed when a user can request a manual report from specific employees by asking a question from the home page, and the system forwards the inquiry to the most appropriate user to provide an answer.	Go to the home page of the MindMerge website, input a question into the system, and verify that the system forwards the inquiry to the appropriate user to provide a manual report.	7	2	2	2	2	2

Id	*Product Name*	*User story*	*Description*	*Acceptance Criteria*	*How to demo*	*Importance*	*Total Estimated Effort*	Remaining Estimated Effort (at the beginning of sprint 1)	Remaining Estimated Effort (at the beginning of sprint 2)	Remaining Estimated Effort (at the beginning of sprint 3)	Remaining Estimated Effort (at the end of sprint 3)
18	Schedule reports	As a manager, I want to define a recurring request for the reports, so that I will be updated regularly on some tasks	From the "Report Page" an user should be able to schedule some reports, such as a specific report answering a specific question is delivered to them every x ammount of time	This task is completed when a manager can define recurring requests for reports from the "Report Page," allowing specific reports to be delivered to them at scheduled intervals.	Go to the "Report Page" on the MindMerge website, select the desired report, specify the frequency (e.g., daily, weekly, monthly), and verify that the system delivers the report answering a specific question to the user at the defined intervals.	8	4	4	4	4	4
19	External notifications	As a user, I want to be able to see major updates at a glance with an external notification system (email), so that I get notified when something important happens	The system shoud be able to send email notification to the users when neaded	This task is completed when the MindMerge system implements an external notification system to send email notifications to users for major updates.	Go to the MindMerge website, perform actions that trigger major updates, and verify that users receive email notifications informing them of the updates.	8	3	3	3	0	0
20	Internal notifications	As a user, I want to be able to see any update withing the platform as I am working on it, so that I get notified when something important happens	The system shoud be able to send in-app norification to the users, that can be viewed from the "Notification View" page	This task is completed when the MindMerge system implements an internal notification system to send in-app notifications to users for any updates within the platform.	Go to the MindMerge website, perform actions that trigger updates, and verify that users receive in-app notifications that can be viewed from the "Notification View" page.	8	6	6	6	6	6
21	Enable/disable notifications	As a user, I want to be able to turn on/off notifications at my will, so that I don't get overwhelmed by useless notifications	An user should be able to enable/disable notification for a specific task from the home page	This task is completed when a user can toggle notifications on or off for specific tasks from the home page.	Go to the home page of the MindMerge website, navigate to the desired task, and verify that users can enable or disable notifications for that task.	5	2	2	2	2	2
22	Testing	As a customer, I want the product to be as relayable and free of bugs as possible, so that it dose not impact my project	The system shoud have automated and manual tests to verify the correctness of the code.	This task is completed when the MindMerge system has both automated and manual tests in place to verify the correctness of the code, ensuring reliability and minimal bugs.	Run the test command and see the test output	10	10	10	5	4	4
23	Deployment	As a customer, I want to receive timely and frequent updates that improve functionalities, so that i can always have the latest features	The system shoud have a mecanism (scripts/github actions) that allows automatic testing and deployment of the new version fo the application	This task is completed when the MindMerge system implements a mechanism, such as scripts or GitHub Actions, for automatic testing and deployment of new versions of the application.	Trigger the deploiment (via script or commit) and verify that the deployment has heppend	10	4	4	0	0	0
24	Database	As a product owner i want my data to be safely stored, and not to risk data losses or data break, so that my project dose not risk to fail because of data losses	The system shoud have a relayable and safe database where to store users data	This task is completed when the MindMerge system utilizes a reliable and secure database to store user data, minimizing the risk of data loss or breaches.	Take a look at the database schema on the MongoDb website	10	5	5	0	0	0

3 Definition of Tests

Test DatabaseManager:TaskManager

This section contains the tests for the TaskManager class, which is responsible for managing tasks in the database.

Test Type	Name	Test case data	Preconditions	Expected Results
Automated	successul task creation	Call the method TaskManager. createTask three times with different task names and check the results.	The TaskModel collection is empty.	A new task is created each time with the expected attributes and status code. The tasks are correctly stored in the database.
Automated	unsuccessul task creation	Call the method TaskManager. createTask with various invalid task data	The TaskModel collection is empty.	Errors.BAD_REQUEST is returned
Automated	successul task note creation	Call the method TaskManager. createTaskNotes and pass a valid taskId and notes	The TaskModel has a few tasks created where notes can be added	Notes for tasks are created
Automated	unsuccessul task note creation	Call the method TaskManager. createTaskNotes with invalid parameters, and/or with non existing task	The TaskModel has a few tasks created where notes can be added	Errors.NOT_FOUND or Errors. BAD_REQUEST is returned based on the invalid parameters
Automated	successful task report schedule creation	Call TaskManager. createTaskReportSchedule with valid report schedule data	The TaskModel has a few tasks created where schedules can be added	A successful status code is returned and the task report schedules are created correctly.
Automated	unsuccessful task report schedule creation	Call TaskManager. createTaskReportSchedule with invalid report schedule data	The TaskModel has a few tasks created where schedules can be added	Errors.NOT_FOUND or Errors. BAD_REQUEST is returned based on the invalid parameters
Automated	successful task update	Call the method TaskManager. updateTask and pass valid parameters	The TaskModel has a few tasks created that can be edited for the test	Tasks are successfully updated with the correct data
Automated	unsuccessful task update	Call the method TaskManager. updateTask and pass non valid parameters	The TaskModel has a few tasks created that can be edited for the test	Errors.NOT_FOUND or Errors. BAD_REQUEST is returned based on the invalid parameters
Automated	successful update last task update	Call TaskManager. updateTaskLastUpdated and pass the id of an existing task	The TaskModel has a few tasks created that can be edited for the test	The task's lastUpdated field is updated correctly, with the current time
Automated	unsuccessful update last task update	Call TaskManager. updateTaskLastUpdated and pass the id of a non existing task	The TaskModel has a few tasks created that can be edited for the test	The expected status code for each update attempt is Errors.NOT_FOUND
Automated	successful update task name	Call the method TaskManager. updateTaskName with valid task and organization IDs, and a new task name	The TaskModel has a few tasks created that can be edited for the test	The task's name is successfuly updated in the database
Automated	unsuccessful update task name	Call the method TaskManager. updateTaskName with invalid task and organization IDs, and/or an invalid name	The TaskModel has a few tasks created that can be edited for the test	Errors.NOT_FOUND or Errors. BAD_REQUEST is returned based on the invalid parameters
Automated	successful update task description	Call the method TaskManager. updateTaskDescription with valid task data	The TaskModel has a few tasks created that can be edited for the test	The task's description is successfuly updated in the database
Automated	unsuccessful update task description	Call the method TaskManager. updateTaskDescription with invalid task data	The TaskModel has a few tasks created that can be edited for the test	Errors.NOT_FOUND or Errors. BAD_REQUEST is returned based on the invalid parameters
Automated	successful update task status	Call the method TaskManager. updateTaskStatus with valid task and status	The TaskModel has a few tasks created that can be edited for the test	The task's status is successfuly updated in the database
Automated	unsuccessful update task status	Call the method TaskManager. updateTaskStatus with invalid task ids and states id	The TaskModel has a few tasks created that can be edited for the test	Errors.NOT_FOUND or Errors. BAD_REQUEST is returned based on the invalid parameters
Automated	successful update task notes	Call the method TaskManager. updateTaskNotes with valid data	The TaskModel has a few tasks created that can be edited for the test	The task's notes are successfuly updated in the database
Automated	unsuccessful update task notes	Call the method TaskManager. updateTaskNotes with invalid data or task id	The TaskModel has a few tasks created that can be edited for the test	Errors.NOT_FOUND or Errors. BAD_REQUEST is returned based on the invalid parameters
Automated	successful add new assaignee	Call the method TaskManager. addNewAssignee with valid assignee ID and task id	The TaskModel has a few tasks created that can be edited for the test	The task's assainees list is successfuly updated in the database
Automated	unsuccessful add new assaignee	Call the method TaskManager. addNewAssignee with invalid assignee ID or task id	The TaskModel has a few tasks created that can be edited for the test	Errors.NOT_FOUND or Errors. BAD_REQUEST is returned based on the invalid parameters
Automated	successful update task manager	Call the method TaskManager. updateTaskManager with a valid new manager, and task id	The TaskModel has a few tasks created that can be edited for the test	The task's manager is successfuly updated in the database
Automated	unsuccessful update task manager	Call the method TaskManager. updateTaskManager with an invalid new manager, or task id	The TaskModel has a few tasks created that can be edited for the test	Errors.NOT_FOUND or Errors. BAD_REQUEST is returned based on the invalid parameters
Automated	successful enable and disable notification	Call TaskManager.enableNotification and TaskManager.disableNotification with valid task IDs	The TaskModel has a few tasks created that can be edited for the test	The task's notification flag is successfuly updated in the database
Automated	unsuccessful enable and disable notification	Call TaskManager.enableNotification and TaskManager.disableNotification with invalid task IDs	The TaskModel has a few tasks created that can be edited for the test	Errors.NOT_FOUND or Errors. BAD_REQUEST is returned based on the invalid parameters
Automated	successful add child task	Call the method TaskManager. addChildTask with a valid task id and child task id	The TaskModel has a few tasks created that can be edited for the test	The task's notes are successfuly updated in the database
Automated	unsuccessful add child task	Call the method TaskManager. addChildTask with an invalid task id or child task id	The TaskModel has a few tasks created that can be edited for the test	Errors.NOT_FOUND or Errors. BAD_REQUEST is returned based on the invalid parameters
Automated	successful update recursive permission value	Call the method TaskManager. updateTaskRecursivePermissionsValue with valid task id and permission value	The TaskModel has a few tasks created that can be edited for the test	The task's recursive permission value issuccessfuly updated in the database
Automated	unsuccessful update recursive permission value	Call the method TaskManager. updateTaskRecursivePermissionsValue with invalid task id or permission value	The TaskModel has a few tasks created that can be edited for the test	Errors.NOT_FOUND or Errors. BAD_REQUEST is returned based on the invalid parameters
Automated	successful delete task	Call the method TaskManager. deleteTask with valid taskId and organizationId	The TaskModel has a few tasks created that can be deleted for the test	the specified task is successfuly deleted in the database
Automated	unsuccessful delete task	Call the method TaskManager. deleteTask with invalid taskId or organizationId	The TaskModel has a few tasks created that can be deleted for the test	Errors.NOT_FOUND or Errors. BAD_REQUEST is returned based on the invalid parameters
Automated	successful delete task notes	Call TaskModel.deleteTaskNote with a valid task and note id	The TaskModel has a few tasks created that can be deleted for the test	the specified task note is successfuly deleted in the database

Automated	unsuccessful delete task notes	Call TaskModel.deleteTaskNote with an invalid task or note id	The TaskModel has a few tasks created that can be deleted for the test	Errors.NOT_FOUND or Errors. BAD_REQUEST is returned based on the invalid parameters
Automated	successful delete task assignee	Call the method TaskManager. deleteTaskAssignee with valid task id, user id, and assignee id	The TaskModel has a few tasks created that can be deleted for the test	the specified task assignee is successfuly deleted from the assignees list in the database
Automated	usuccessful delete task assignee	Delete task assignee with invalid task id, assignee id, or organization id	The TaskModel has a few tasks created that can be deleted for the test	Errors.NOT_FOUND or Errors. BAD_REQUEST is returned based on the invalid parameters
Automated	successful delete report schedule	Delete report schedule with valid task id, report schedule id, and organization id	The TaskModel has a few tasks created that can be deleted for the test	the specified task report schedule is successfuly deleted in the database
Automated	unsuccessful delete report schedule	Delete report schedule with invalid task id, report schedule id, or organization id	The TaskModel has a few tasks created that can be deleted for the test	Errors.NOT_FOUND or Errors. BAD_REQUEST is returned based on the invalid parameters
Automated	successful remove child task	Remove child task with valid parent task id, child task id, and organization id	The TaskModel has a few tasks created that can be deleted for the test	the specified child task is successfuly deleted from the child tasks list in the database
Automated	unsuccessful remove child task	Remove child task with invalid parent task id, child task id, or organization id	The TaskModel has a few tasks created that can be deleted for the test	Errors.NOT_FOUND or Errors. BAD_REQUEST is returned based on the invalid parameters
Automated	successful get task	Get task with valid task id and organization id	The TaskModel has a few tasks created that can be red for the test	Successfully retrieve task details, returning Errors.OK
Automated	unsuccessful get task	Get task with invalid task id or organization id	The TaskModel has a few tasks created that can be red for the test	Task not found, returning Errors. NOT_FOUND

Test OrganizationManager

This section is dedicated to the tests for the OrganizationManager class, which is responsible for managing organizations in the database, with the integrated API.

Automated	test successfull add user to organization API	Call the API to add an user to the organization with a valid requeat	User and Organizations are already present in the database	The api returns OK (200) and the database is updated
Automated	test unsuccesfull add user to organization API	Call the API to add an user to the organization with invalid users/organizations	User and Organizations are already present in the database	Not found error (404) or Bad request error (400) is returned based on the invalid parameters
Automated	test unautorized add user to organization API	Call the API to add an user to the organization with invalid/missing autorization token	User and Organizations are already present in the database	Non autorized error (403)
Automated	test successfull remove user from organization API	Call the API to remove an user from an organization with a valid requeat	User and Organizations are already present in the database	The api returns OK (200) and the database is updated
Automated	test unsuccesfull remove user from organization API	Call the API to remove an user from an organization with invalid users/organizations	User and Organizations are already present in the database	Not found error (404) or Bad request error (400) is returned based on the invalid parameters
Automated	test unautorized remove user from organization API	Call the API to remove an user from an organization with invalid/missing autorization token	User and Organizations are already present in the database	Non autorized error (403)
Automated	test successful calculate subscription price	Call the method OrganizationManager. calculateSubscriptionPrice with a valid organization id	an Organization with some users and soem tasks is present in the database	The function successfuly caluclate the price the ortanization should pay based on usage
Automated	test unsuccessful calculate subscription price	Call the method OrganizationManager. calculateSubscriptionPrice with an invalid organization id	an Organization with some users and soem tasks is present in the database	Errors.NOT_FOUND or Errors. BAD_REQUEST is returned based on the invalid parameters
Automated	test succesful verify subscription	Call the method OrganizationManager. verifySubscription with a valid organization id	an Organization is present in the database	The function verify the current status of the organization subscription and returns it
Automated	test unsuccesful verify subscription	Call the method OrganizationManager. verifySubscription with an invalid organization id	an Organization is present in the database	Errors.NOT_FOUND or Errors. BAD_REQUEST is returned based on the invalid parameters
Automated	test successful pay subscription	Call the API to pay the subscription with valid organization, and credit card informations	an Organization is present in the database, A connection with a bancking service has been established, A valid tast credit card is ready	The payment is successful
Automated	test unsuccessful pay subscription	Call the API to pay the subscription with invalid organization, or credit card informations	an Organization is present in the database, A connection with a bancking service has been established, A valid tast credit card is ready	The payment is not successful

Test Report Page

This section includes manual tests for the Report Page, which is responsible for generating reports.

Manual	test add periodic report	Go to the report page and take a look at the list of scheduled reports. Then add a few new repors by using the button "Add report schedule"	The backend, the front end and the database are running. You have access to a testing account	The added report are displayed in the table and all the informations are corrects. This is true even after a log out and a browser cash refresh
Manual	test delete periodic report	Go to the report page and take a look at the list of scheduled reports. Then click the delete button on some of them.	The backend, the front end and the database are running. You have access to a testing account	The deleted reports are no longer displayed in the list This is true even after a log out and a browser cash refresh. The user no longer receive notifications about the scheduler reports

Test NotificationManager

This section includes automated tests for the NotificationManager class, which is responsible for managing notifications in the database.

Automated	test successfull send notification	Call method NotificationManager. sendNotification with valid parameters	At least one user and one notification are present in the database	The notification is sent successfully
Automated	test unsuccessfull send notification	Call method NotificationManager. sendNotification with invalid parameters	At least one user and one notification are present in the database	Errors.NOT_FOUND or Errors. BAD_REQUEST is returned based on the invalid parameters
Automated	test successfull send notification	Call method ExternalNotificationManager. sendNotification with valid parameters	At least one user and one notification are present in the database, we have a mail account with api access to test the notification	The notification is sent successfully
Automated	test unsuccessfull send notification	Call method ExternalNotificationManager. sendNotification with invalid parameters	At least one user and one notification are present in the database, we have a mail account with api access to test the notification	Errors.NOT_FOUND or Errors. BAD_REQUEST is returned based on the invalid parameters
Automated	test successfull delete notification	Call method InternalNotificationManager. deleteNotification with valid parameters (userId, notificationId, userToken)	At least one user and one notification are present in the database	The notification is deleted successfully
Automated	test unsuccessfull delete notification	Call method InternalNotificationManager. deleteNotification with invalid parameters (userId, notificationId, userToken)	At least one user and one notification are present in the database	Errors.NOT_FOUND or Errors. BAD_REQUEST is returned based on the invalid parameters
Automated	test successfull mark notification as read	Call method InternalNotificationManager. markNotificationAsRead with valid parameters (userId, notificationId, userToken)	At least one user and one notification are present in the database	The notification gets marked as read
Automated	test unsuccessfull mark notification as read	Call method InternalNotificationManager. markNotificationAsRead with invalid parameters (userId, notificationId, userToken)	At least one user and one notification are present in the database	Errors.NOT_FOUND or Errors. BAD_REQUEST is returned based on the invalid parameters
Automated	test successfull get notification details	Call method InternalNotificationManager. getNotificationDetails with valid parameters (userId, notificationId, userToken)	At least one user and one notification are present in the database	A notification object is returned
Automated	test unsuccessfull get notification details	Call method InternalNotificationManager. getNotificationDetails with invalid parameters (userId, notificationId, userToken)	At least one user and one notification are present in the database	Errors.NOT_FOUND or Errors. BAD_REQUEST is returned based on the invalid parameters
Automated	test successfull get notification list	Call method InternalNotificationManager. getNotificationDetails with valid parameters (userId, userToken)	At least one user is present in the database	The list of notifications linked to the userId are returned
Automated	test unsuccessfull get notification list	Call method InternalNotificationManager. getNotificationDetails with invalid parameters (userId, userToken)	At least one user is present in the database	Errors.BAD_REQUEST is returned

Test ReportManager

This section includes automated tests for the ReportManager class, which is responsible for managing reports in the database.

Automated	test successfull set new report schedule	Call method reportScheduler. scheduleReport with valid parameters (organizationId, taskId, reportPrompt, reportKind, reportFrequency, userId, userToken)	Organization, Task, and User are already present in the database	A new report schedule is set for said task and with said frequency
Automated	test unsuccessfull set new report schedule	Call method reportScheduler. scheduleReport with invalid parameters (organizationId, taskId, reportPrompt, reportKind, reportFrequency, userId, userToken)	Organization, Task, and User are already present in the database	Errors.NOT_FOUND or Errors. BAD_REQUEST is returned based on the invalid parameters
Automated	test successfull get report schedules	Call method reportScheduler. getReportSchedules with valid parameters (organizationId, taskId, userId, userToken)	Organization, Task, and User are already present in the database	The list of report schedules set for the task are returned
Automated	test unsuccessfull get report schedules	Call method reportScheduler. getReportSchedules with invalid parameters (organizationId, taskId, userId, userToken)	Organization, Task, and User are already present in the database	Errors.NOT_FOUND or Errors. BAD_REQUEST is returned based on the invalid parameters
Automated	test successfull remove report schedule	Call method reportScheduler. deleteReportSchedule with valid parameters (organizationId, taskId, reportId, userId, userToken)	Organization, Task, User and report are already present in the database	The report schedule gets deleted successfully
Automated	test unsuccessfull remove report schedule	Call method reportScheduler. deleteReportSchedule with invalid parameters (organizationId, taskId, reportId, userId, userToken)	Organization, Task, User and report are already present in the database	Errors.NOT_FOUND or Errors. BAD_REQUEST is returned based on the invalid parameters
Automated	test execute all pending periodic reports	Call method reportScheduler. executeScheduledReport with valid parameters (organizationId)	An Organization is present in the database	All the pending reports gets executed
Automated	test unsuccessfull execute all pending periodic reports	Call method reportScheduler. executeScheduledReport with invalid parameters (organizationId)	An Organization is present in the database	Errors.BAD_REQUEST is returned
Automated	test successfull generate automatic report	Call method AutomaticReportManager. generateAutomaticReports with valid parameters (organizationId, taskId, reportPrompt)	Organization and Task are already present in the database, connection with LLM APIs is valid	The report gets generated successfully
Automated	test unsuccessfull generate automatic report	Call method AutomaticReportManager. generateAutomaticReports with invalid parameters (organizationId, taskId, reportPrompt)	Organization and Task are already present in the database, connection with LLM APIs is valid	Errors.NOT_FOUND or Errors. BAD_REQUEST is returned based on the invalid parameters
Automated	test successfull generate manual report	Call method ManualReportManager. generateManualReports with valid parameters (organizationId, taskId, reportPrompt)	Organization and Task are already present in the database	The report gets generated successfully
Automated	test unsuccessfull generate manual report	Call method ManualReportManager. generateManualReports with invalid parameters (organizationId, taskId, reportPrompt)	Organization and Task are already present in the database	Errors.NOT_FOUND or Errors. BAD_REQUEST is returned based on the invalid parameters
Automated	test successfull generate report	Call method ReportManager. generateReport with valid parameters (organizationId, taskId, reportPrompt, reportKind, userId, userToken)	Organization, Task and User are already present in the database, connection with LLM APIs is valid	The report gets generated successfully
Automated	test unsuccessfull generate report	Call method ReportManager. generateReport with invalid parameters (organizationId, taskId, reportPrompt, reportKind, userId, userToken)	Organization, Task and User are already present in the database, connection with LLM APIs is valid	Errors.NOT_FOUND or Errors. BAD_REQUEST is returned based on the invalid parameters

Test DatabaseManager::OrganizationManager

This section is dedicated to the tests for the OrganizationManager class, which is responsible for managing organizations in the database.

Automated	successful organization creation	Call the method OrganizationManager. createOrganization with valid data and check the result	OrganizationModel database is empty	New organization is successfully created with provided data, and all attributes are correctly saved in the database
Automated	invalid organization creation	Call the method OrganizationManager. createOrganization with various invalid data and check the results	OrganizationModel database is empty	The response status code should be Errors. BAD_REQUEST for each invalid organization data combination.
Automated	successful add user to organization	Call the method OrganizationManager. addUserToOrganization with valid input and check the result	Organization and User should already exist in the database	Errors.OK is returned. The user should be successfully added to the organization, and the organization's user list should include the added user ID.
Automated	unsuccessful add user to organization	Call the method OrganizationManager. addUserToOrganization with invalid input and check the result		Invalid attempts should return Errors. NOT_FOUND or Errors.BAD_REQUEST
Automated	update license	Call the method OrganizationManager. updateLicense and check the results	An organization with the specified ID exists in the database	Valid attempts should return Errors.OK and license is updated. Invalid attempts should return Errors.NOT_FOUND or Errors. BAD_REQUEST
Automated	update license expiration date	Call the method OrganizationManager. updateLicenseExpirationDate and check the results	An organization with the specified ID exists in the database	Valid attempts should return Errors.OK and date is updated. Invalid attempts should return Errors.NOT_FOUND or Errors. BAD_REQUEST
Automated	read organization	Call the method OrganizationManager. readOrganization and check the results	An organization with the specified ID exists in the database	The response status code should be Errors. OK. The organization data should match the those created earlier.

Test AccountManager

This section includes automated tests for the AccountManager class, which is responsible for managing accounts in the database.

Automated	test successful find user by name	Call the method UserManager:: readUserByName with valid input and check the result	A user with the specified name exists in the database	The response status code should be Errors. OK.
Automated	test unsuccessful find user by name	Call the method UserManager:: readUserByName with invalid input and check the result		The response status code should be Errors. NOT_FOUND or Errors.BAD_REQUEST depending on the incorrect imput inserted
Automated	test getGoogleOauthLogInInfo	Call the method UserManager:: readGoogleUserInfo	The specified user should exist in the database and have performed signup with Google	The user infos are returned correctly if the user exists
Automated	test getFacebookOauthLogInInfo	Call the method UserManager:: readFacebookUserInfo	The specified user should exist in the database and have performed signup with Facebook	The user infos are returned correctly if the user exists
Automated	test successful custom login	Perform user login using correct username and password	The specified user should exist in the database	The user is correctly logged in
Automated	test unsuccessful custom login	Perform user login trying to use incorrect parameters	The specified user should not exist in the database or the password should be incorrect	The response status code should be Errors. NOT_FOUND or Errors.BAD_REQUEST and logging in is not performed
Automated	test successful Google login	Perform user login using Google	The specified user should exist in the database and have performed signup with Google	The user is correctly logged in using Google
Automated	test unsuccessful Google login	Perform user login using Google inserting invalid credentials	The specified user should not exist in the database or not having performed signup with Google	The response status code should be Errors. NOT_FOUND or Errors.BAD_REQUEST and logging in is not performed
Automated	test successful Facebook login	Perform user login using Facebook	The specified user should exist in the database and have performed signup with Facebook	The user is correctly logged in using Facebook
Automated	test unsuccessful Facebook login	Perform user login using Facebook inserting invalid credentials	The specified user should not exist in the database or not having performed signup with Facebook	The response status code should be Errors. NOT_FOUND or Errors.BAD_REQUEST and logging in is not performed
Automated	test successful custom signup	Call the method UserManager:: createUser with valid input and check the result	The UserModel collection is empty	The response status code should be Errors. OK and the user is correctly created
Automated	test unsuccessful custom signup	Call the method UserManager:: createUser with invalid input and check the result	The UserModel collection is empty	The response status code should be Errors. BAD_REQUEST and an error message should be returned
Automated	test Google signup	Perform user signup using Google	The UserModel collection is empty	True is returned if the user is created successfully, otherwise False is returned
Automated	test Facebook signup	Perform user signup using Facebook	The UserModel collection is empty	True is returned if the user is created successfully, otherwise False is returned
Automated	test successful edit username	Call the method UserManager:: updateUserName with valid input	The user should exist in the database	The response status code should be Errors. OK and the username is updated correctly
Automated	test unsuccessful edit username	Call the method UserManager:: updateUserName with invalid input		The response status code should be Errors. NOT_FOUND or Errors.BAD_REQUEST, the username is not updated
Automated	test successful change password	Change user password with correct old password and valid new one	The user should exist in the database and the password inserted should correspond to the original one	The response status code should be Errors. OK and the password is updated
Automated	test unsuccessful change password	Change user password with incorrect old password or invalid new one	The user should exist in the database	The response status code should be Errors. BAD_REQUEST and an error message should be returned
Automated	test successful delete account	Call the method UserManager:: deleteAccount with valid input and check the result	The account should exist in the database	The response status code should be Errors. OK and the account is correctly deleted
Automated	test unsuccessful delete account	Call the method UserManager:: deleteAccount with invalid input and check the result	The account should not exist in the database	The response status code should be Errors. NOT_FOUND and an error message is displayed

Test DatabaseManager::UserManager

This section is dedicated to the tests for the UserManager class, which is responsible for managing users in the database.

Automated	test successful create user	Call the method UserManager. createUser with several correct inputs and check the result	The UserModel collection is empty	The response status code should be Errors. OK and the user is correctly created
Automated	test unsuccessful create user	Call the method UserManager. createUser with several invalid inputs and check the result	The UserModel collection is empty	The response status code should be Errors. BAD_REQUEST and an error message is displayed
Automated	test successful find user	Call the method UserManager.findUser for an existing user and check the result	The UserModel collection should contain the user	The response status code should be Errors. OK and the user is correctly returned
Automated	test unsuccessful find user by name	Call the method UserManager.findUser for a non existing and/or with invalid parameters	The UserModel collection should not contain the user	The response status code should be Errors. NOT_FOUND or Errors.BAD_REQUEST and an error message is displayed
Automated	test successful find every user	Call the method UserManager. findEveryUser and check the results	The UserModel collection should be non-empty	The response status code should be Errors. OK and every user is correctly found
Automated	test successful create notification	Call the method UserManager. createNotification with invalid inputs and an existing user and check the result	The corrisponding user should be contained the UserModel collection	The response status code should be Errors. OK and the notification is correctly created
Automated	test unsuccessful crete notification	Call the method UserManager. createNotification with invalid inputs and/or a non existing user	The corrisponding user should not be contained the UserModel collection and/or the input for the method should be invalid	The response status code should be Errors. NOT_FOUND or Errors.BAD_REQUEST and an error message is displayed
Automated	test successful read notification	Call the method UserManager. readNotification for an existing user and check the result	The related user should already be contained in the UserModel collection	The response status code should be Errors. OK and the notification is returned
Automated	test unsuccessful read notification	Call the method UserManager. readNotification for a non existing user and check the result	The corrisponding user should not be contained the UserModel collection and/or the input for the method should be invalid	The response status code should be Errors. NOT_FOUND or Errors.BAD_REQUEST and an error message is displayed
Automated	test successful mark notification as read	Call the method UserManager. MarkNotificationAsRead for an existing user and a valid notification Id and check the result	The related user should already be contained in the UserModel collection and the related notification should already exist	The response status code should be Errors. OK and the flag for the notification is correctly updated
Automated	test unsuccessful mark notification as read	Call the method UserManager. MarkNotificationAsRead for a non existing notification id and check the result	Either the user or the notification should be not contained in their corrisponding collection	The response status code should be Errors. NOT_FOUND or Errors.BAD_REQUEST and an error message is displayed
Automated	test successful update username	Call the method UserManager. updateUserName with a valid new username for an existing user and check the result	The user should be already contained in the UserModel collection and the new username shoud be valid and not already taken	The response status code should be Errors. OK and the username is correctly updated
Automated	test unsuccessful update username	Call the method UserManager. updateUserName with an invalid new username and/or for a non existing user and check the result	The user should not be already contained in the UserModel collection and/or the new username should be invalid/already taken	The response status code should be Errors. NOT_FOUND or Errors.BAD_REQUEST and an error message is displayed
Automated	test successful update user kind	Call the method UserManager. updateUserKind with a valid user kind for an existing user and check the result	The user should be already contained in the UserModel collection and the new user kind shoud be valid and not already taken	The response status code should be Errors. OK and the user kind is correctly updated
Automated	test unsuccessful update user kind	Call the method UserManager. updateUserKInd with an invalid user kind and/or for a non existing user and check the result	The user should not be already contained in the UserModel collection and/or the new user kind should be invalid	The response status code should be Errors. NOT_FOUND or Errors.BAD_REQUEST and an error message is displayed
Automated	test successful update user mail	Call the method UserManager. updateUserMail with valid new mail for an existing user and check the result	The user should be already contained in the UserModel collection and the new mail shoud be valid and not already taken	The response status code should be Errors. OK and the user email is correctly updated
Automated	test unsuccessful update user mail	Call the method UserManager. updateUserMail with an invalid new mail and/or for a non existing user and check the result	The user should not be already contained in the UserModel collection and/or the new email should be invalid/already taken	The response status code should be Errors. NOT_FOUND or Errors.BAD_REQUEST and an error message is displayed

Test TaskManager

This section includes automated tests for the TaskManager class, which is responsible for managing tasks in the database.

Automated	test successful create task	Call the method TaskCreator.createTask with valid input and existing related collections	The user token should be valid and the inputs should be valid and existing in their corrisponding collection	The response status code should be Errors.OK and the task is correctly created
Automated	test unsuccessful create tak	Call the method TaskCreator.createTask with invalid input and/or one or more non-existing related collection	The user token should not be valid and/or the inputs should be invalid or non existing in their corrisponding collection	The response status code should be Errors.NOT_FOUND or Errors.BAD_REQUEST and an error message is displayed
Automated	test successful create task notes	Call the method TaskCreator.createTasNoteswith valid input and existing related collections	The user token should be valid and the inputs should be valid and existing in their corrisponding collection	The response status code should be Errors.OK and the task notes are correctly created
Automated	test unsuccessful create task notes	Call the method TaskCreator.createTaskNotes with invalid input and/or one or more non-existing related collection	The user token should not be valid and/or the inputs should be invalid or non existing in their corrisponding collection	The response status code should be Errors.NOT_FOUND or Errors.BAD_REQUEST and an error message is displayed
Automated	test successful delete task	Call the method TaskEditor.deleteTask with valid input and existing related collections	The user token should be valid and the inputs should be valid and existing in their corrisponding collection	The response status code should be Errors.OK and the task is correctly deleted
Automated	test unsuccessful delete task	Call the method TaskEditor.deleteTask with invalid input and/or one or more non-existing related collection	The user token should not be valid and/or the inputs should be invalid or non existing in their corrisponding collection	The response status code should be Errors.NOT_FOUND or Errors.BAD_REQUEST and an error message is displayed
Automated	test successful delete task assignee	Call the method TaskEditor.deleteTaskAssignee with valid input and existing related collections	The user token should be valid and the inputs should be valid and existing in their corrisponding collection	The response status code should be Errors.OK and the task assignee is correctly deleted
Automated	test unsuccessful delete task assignee	Call the method TaskEditor.deleteTaskAssignee with invalid input and/or one or more non-existing related collection	The user token should not be valid and/or the inputs should be invalid or non existing in their corrisponding collection, assigness should be 2 at least	The response status code should be Errors.NOT_FOUND or Errors.BAD_REQUEST and an error message is displayed
Automated	test successful delete task notes	Call the method TaskEditor.deleteTaskNotes with valid input and existing related collections	The user token should be valid and the inputs should be valid and existing in their corrisponding collection	The response status code should be Errors.OK and the task notes are correctly deleted
Automated	test unsuccessful delete task notes	Call the method TaskEditor.deleteTaskNotes with invalid input and/or one or more non-existing related collection	The user token should not be valid and/or the inputs should be invalid or non existing in their corrisponding collection	The response status code should be Errors.NOT_FOUND or Errors.BAD_REQUEST and an error message is displayed
Automated	test successful update task	Call the method TaskEditor.updateTask with valid input and existing related collections	The user token should be valid and the inputs should be valid and existing in their corrisponding collection	The response status code should be Errors.OK and the task is correctly updated
Automated	test unsuccessful update task	Call the method TaskEditor.updateTask with invalid input and/or one or more non-existing related collection	The user token should not be valid and/or the inputs should be invalid or non existing in their corrisponding collection	The response status code should be Errors.NOT_FOUND or Errors.BAD_REQUEST and an error message is displayed
Automated	test successful update task name	Call the method TaskEditor.updateTaskName with valid input and existing related collections	The user token should be valid and the inputs should be valid and existing in their corrisponding collection	The response status code should be Errors.OK and the task name is correctly updated
Automated	test unsuccessful update task name	Call the method TaskEditor.updateTaskName with invalid input and/or one or more non-existing related collection	The user token should not be valid and/or the inputs should be invalid or non existing in their corrisponding collection	The response status code should be Errors.NOT_FOUND or Errors.BAD_REQUEST and an error message is displayed
Automated	test successful update task description	Call the method TaskEditor.updateTaskDescription with valid input and existing related collections	The user token should be valid and the inputs should be valid and existing in their corrisponding collection	The response status code should be Errors.OK and the task description is correctly updated
Automated	test unsuccessful update task description	Call the method TaskEditor.updateTaskDescription with invalid input and/or one or more non-existing related collection	The user token should not be valid and/or the inputs should be invalid or non existing in their corrisponding collection	The response status code should be Errors.NOT_FOUND or Errors.BAD_REQUEST and an error message is displayed
Automated	test successful update task status	Call the method TaskEditor.updateTaskStatus with valid input and existing related collections	The user token should be valid and the inputs should be valid and existing in their corrisponding collection	The response status code should be Errors.OK and the task status is correctly updated
Automated	test successful update task status	Call the method TaskEditor.updateTaskStatus with invalid input and/or one or more non-existing related collection	The user token should not be valid and/or the inputs should be invalid or non existing in their corrisponding collection	The response status code should be Errors.NOT_FOUND or Errors.BAD_REQUEST and an error message is displayed
Automated	test successful update task notes	Call the method TaskEditor.updateTaskNotes with valid input and existing related collections	The user token should be valid and the inputs should be valid and existing in their corrisponding collection	The response status code should be Errors.OK and the task notes are correctly updated
Automated	test successful update task notes	Call the method TaskEditor.updateTaskNotes with invalid input and/or one or more non-existing related collection	The user token should not be valid and/or the inputs should be invalid or non existing in their corrisponding collection	The response status code should be Errors.NOT_FOUND or Errors.BAD_REQUEST and an error message is displayed
Automated	test successful update task assignee	Call the method TaskEditor.updateTaskAssignee with valid input and existing related collections	The user token should be valid and the inputs should be valid and existing in their corrisponding collection	The response status code should be Errors.OK and the task assignee is correctly updated
Automated	test unsuccessful update task assignee	Call the method TaskEditor.updateTaskAssignee with invalid input and/or one or more non-existing related collection	The user token should not be valid and/or the inputs should be invalid or non existing in their corrisponding collection	The response status code should be Errors.NOT_FOUND or Errors.BAD_REQUEST and an error message is displayed
Automated	test successful add new assignee	Call the method TaskEditor.addNewAssignee with valid input and existing related collections	The user token should be valid and the inputs should be valid and existing in their corrisponding collection, user should not yet belong to it	The response status code should be Errors.OK and a task tree array is correctly returned

Automated	test unsuccessful add new assignee	Call the method TaskEditor. addNewAssignee with invalid input and/or one or more non-existing related collection	The user token should not be valid and/or the inputs should be invalid or non existing in their corrisponding collection or user already belongs to it	The response status code should be Errors. NOT_FOUND or Errors.BAD_REQUEST and an error message is displayed
Automated	test successful update task manager	Call the method TaskEditor. updateTaskManager with valid input and existing related collections	The user token should be valid and the inputs should be valid and existing in their corrisponding collection	The response status code should be Errors. OK and the manager for the task is correctly updated
Automated	test unsuccessful update task manager	Call the method TaskEditor. updateTaskManager with invalid input and/or one or more non-existing related collection	The user token should not be valid and/or the inputs should be invalid or non existing in their corrisponding collection	The response status code should be Errors. NOT_FOUND or Errors.BAD_REQUEST and an error message is displayed
Automated	test unsuccessful enable notification	Call the method TaskEditor. enableNotification with existing collections as input	The user token should be valid and the inputs should be valid and existing in their corrisponding collection	The response status code should be Errors. OK and the flag is coretly modified
Automated	test unsuccessful enable notification	Call the method TaskEditor. enableNotification with non-existing collections as input	The user token should not be valid and/or the inputs should be invalid or non existing in their corrisponding collection	The response status code should be Errors. NOT_FOUND or Errors.BAD_REQUEST and an error message is displayed
Automated	test successful disable notification	Call the method TaskEditor. disableNotification with existing collections as input	The user token should be valid and the inputs should be valid and existing in their corrisponding collection	The response status code should be Errors. OK and the flag is coretly modified
Automated	test unsuccessful disable notification	Call the method TaskEditor. disableNotification with non-existing collections as input	The user token should not be valid and/or the inputs should be invalid or non existing in their corrisponding collection	The response status code should be Errors. NOT_FOUND or Errors.BAD_REQUEST and an error message is displayed
Automated	test successful update task recursive permissions	Call the method TaskEditor. updateTaskRecursivePermissionsValue with existing collections as input	The user token should be valid and the inputs should be valid and existing in their corrisponding collection	The response status code should be Errors. OK and the permissions are correctly modified
Automated	test unsuccessful update task recursive permissions	Call the method TaskEditor. updateTaskRecursivePermissionsValue with non-existing collections as input and/or invalid input	The user token should not be valid and/or the inputs should be invalid or non existing in their corrisponding collection	The response status code should be Errors. NOT_FOUND or Errors.BAD_REQUEST and an error message is displayed
Automated	test successful get tasks for user	Call the method TaskGetter. getTasksForUser with valid input and existing related collections	The user token should be valid and the inputs should be valid and existing in their corrisponding collection	The response status code should be Errors. OK and the tasks for the user are correctly returned
Automated	test unsuccessful get tasks for user	Call the method TaskGetter. getTasksForUser with invalid input and/or one or more non-existing related collection	The user token should not be valid and/or the inputs should be invalid or non existing in their corrisponding collection	The response status code should be Errors. NOT_FOUND or Errors.BAD_REQUEST and an error message is displayed
Automated	test successful get task id	Call the method TaskGetter.getTask with valid input and existing related collections	The user token should be valid and the inputs should be valid and existing in their corrisponding collection	The response status code should be Errors. OK and the task is correctly returned
Automated	test unsuccessful get task id	Call the method TaskGetter.getTask with invalid input and/or one or more non-existing related collection	The user token should not be valid and/or the inputs should be invalid or non existing in their corrisponding collection	The response status code should be Errors. NOT_FOUND or Errors.BAD_REQUEST and an error message is displayed

Test Home Page

This section includes manual tests for the home page.

Manual	test successful get user's tasks	Go to the home page and click on the task related button, then click on the "list" related button	A valid session is being running and both front-end and back-end are correcly working	The task list is correctly displayed
Manual	test successful get task visualization info	Go to the home page and click on the task related button, then click on the "visualize" related button	A valid session is being running and both front-end and back-end are correcly working	The task tree hierarchy is correctly displayed
Manual	test successful create task	Go to the home page and click on the task related button, then click on the "create new one" related button	A valid session is being running and both front-end and back-end are correcly working	The new task is correctly added and the list is updated
Manual	test successful delete task	Go to the home page and click on the task related button, select a task and then click on the "delete" related button	A valid session is being running and both front-end and back-end are correcly working	The selected task is correctly removed and is no longer displayed on the corrisponding list
Manual	test successful edit task	Go to the home page and click on the task related button, select a task and then click on the "list" related button	A valid session is being running and both front-end and back-end are correcly working	The selected task is correctly modified and the new attributes are correctly displayed in the corrisponding list

Test Notification View

This section includes manual tests for the notification view.

Manual	test delete notification	Go to notification view and see the list of notifications. Then click on one of them and select the delete option	At least one user and one notification should exist in the database	Notifications are deleted and no longer displayed in the notifications list
Manual	test mark notification as read	Go to notification view and see the list of notifications. Then click on one of them and mark it as read	At least one user and one unread notification should exist in the database	The notification is marked as read but is still displayed in the notifications list
Manual	test get notifications list	Go to notification view and click on notification list to see the list of all notifications	At least one user should exist in the database	The notification list is displayed. If there are no notifications, an empty list should be displayed
Manual	test get notifications details	Go to notification view and see the list of notifications. Then click on one of them and select get details	At least one user and one notification should exist in the database	A new page is displayed with all the details about the selected notification (Notification and user ids, notification text, date and read flag)

Test Sign In and Sign Up Pages

This section includes manual tests for the sign in and sign up pages.

Manual	test successful sign in	Go to the sign in page and insert valid parameters as username and password, then click on the sign in button	The user must already exist in the database but not be already signed in. The credentials should be the correct ones associated to that user	The user is correctly logged in and is taken to the home page
Manual	test unsuccessful sign in	Go to the sign in page and insert incorrect parameters as username and password, then click on the sign in button	The user must not be already signed in	The user is not logged in and an error message is displayed. The user is taken back to the login page
Manual	test successful sign up	Go to the sign up page and insert all the requested valid parameters, then click on the sign up button	The user should not already exist in the database. The parameters should be valid	The user is correctly siged up and is taken to the home page
Manual	test unsuccessful sign up	Go to the sign up page and insert invalid parameters or leave some parameters blank, then click on the sign up button	The user should not already exist in the database	The user is not signed up and an error message is displayed, saying if there are missing parameters or invalid ones. The user is taken back to the sign up page

Test Edit Profile Page

This section includes manual tests for the edit profile page.

Manual	test successful change password	Go to the edit profile page, select change password, in the first textbox insert the old password and then in the second textbox the new password	The user must be already signed in	The user is redirected to the edit profile page, the new password is set
Manual	test unsuccessful change password	Go to the edit profile page, select change password, in the first textbox insert a wrong password and then in the second textbox the new password	The user must be already signed in	The user gets a wrong password message and the page is reloaded.
Manual	test usuccessful change password	Go to the edit profile page, select change password, in the first textbox insert the old password and then in the second textbox an old password	The user must be already signed in	The user gets a "You can't use an old password" message, the page is reloaded
Manual	test successful delete user account	Go to the edit profile page, select delete account, insert your password and then click on the confirm button	The user must be already signed in	The user is redirected to the home page, the account has been deleted successfully
Manual	test unsuccessful delete user account	Go to the edit profile page, select delete account, insert a wrong password and then click on the confirm button	The user must be already signed in	The user gets a wrong password message and the page is reloaded.

Test Organization View Page

This section includes manual tests for the organization view page.

Manual	test add user to organization	Go to the Organization view page, select add user to organization, search the new user to add, select it and click on the confirm button	Organization and new User must be already defined in the database. The Organization manager must be signed in	The new user is added into the Organization, Organization manager is redirected into Organization view page
Manual	test delete Organization	Go to the Organization view page, select delete Organization button, insert password, and click to confirm button	Organization must be already defined in the database. The Organization manager must be signed in	The Owner gets redirected into the home page and the Organization gets removed from the database, all task and reports included
Manual	test unsuccessful delete Organization	Go to the Organization view page, select delete Organization button, insert wrong password, and click to confirm button	Organization must be already defined in the database. The Organization manager must be signed in	The Owner gets a wrong password message and the page gets reloaded
Manual	test successful change payment method	Go to the Organization view page, select change payment method, select the payment method of choise, authenticate into new selected bank account, confirm	Organization must be already defined in the database. The Organization manager must be signed in, connection with the bank must be authenticated	The Owner gets redirected into the Organization view page, new payment method is set
Manual	test unsuccessful change payment method	Go to the Organization view page, select change payment method, select the payment method of choise, fail to authenticate into new selected bank account	Organization must be already defined in the database. The Organization manager must be signed in	The Owner gets redirected into the Organization view page, all changes are reverted

4 Git Strategy

Among the several git strategies available and commonly used, we have chosen **GitHub Flow**. We selected this strategy because it is *one of the simplest* and is particularly suitable for small teams like ours. GitHub Flow's straightforward approach, centered around a single branch with short-lived feature branches, allows us to maintain a **clean and manageable codebase**. This strategy also facilitates *continuous integration and delivery*, making it easier to track progress and integrate changes seamlessly.

5 Sprint 1

5.1 Backlog

					New estimates of effort (remaining at the end of day ...)						
Sprint task	**Product backlog items**	**Volunteer**	**DoD**	**Initial estimate of effort**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Setup testing infrastructure	As a customer, I want the product to be as relayable and free of bugs as possible	Everyone	The testing is setup in such a way that we can verify how many of the tests fail\pass just by running a command in the terminal	3	2	1	0	0	0	0	0
Setup Mongodb	As a product owner i want my data to be safely stored, and not to risk data losses or data breaks	Luca S.	A schema for every class required is defined	3	2	1	0	0	0	0	0
Add function to create/edit/delete tasks in the database + testing	As a user, I want to be able to freely create a task at any time	Luca S.	Every functionality necessary for managing the tasks is implemented	8	8	6	5	4	2	0	0
	As a user, I want to be able to edit the task (e.g. title, description etc)										
	As a user, I want to be able to delete the task at any given time										
Add function to create/edit/delete users in the database + testing	As a user, I want to be able to create an account	Gioele	Every functionality necessary for managing the users is implemented	4	4	4	3	3	2	1	0
	As a user, I want to be able to edit my account information										
	As a user, I want be able to delete my own account at any moment										
Add function to create/edit/delete organizations in the database + testing	As a manager, since I want to use the platform, I want to be able to create an organization at any moment	Luca F.	Every method for the organization manager is implemented	4	4	4	3	3	2	1	0
	As an owner, as my organization grows, I want to be able to add a user at any time										
	As an owner, I want to be able to remove a user within my organization at any time										
Set up CI/CD scripts	As a customer, I want to receive timely and frequent updates that improve functionalities	Gabriele	All necessary scripts are well-written and are proven to work	3	3	3	2	2	1	1	0

5.2 Review and Retrospective

5.2.1 Retrospective

Overall, the sprint was successful, as we managed to complete all the tasks on time.

We experienced some minor slowdowns due to our unfamiliarity with *JavaScript* and *MongoDB*, but we ultimately overcame these challenges. This sprint has been a valuable learning experience, highlighting areas where we can improve our technical skills.

In terms of task estimation, our decision to adopt a **conservative approach** proved to be beneficial. The higher estimates allowed us to allocate sufficient time to each task, reducing stress and enabling us to handle unforeseen issues effectively.

Team coordination was commendable, though there is room for improvement. The meticulous division of labor contributed significantly to our success. Each member had a clear understanding of their responsibilities, which facilitated smoother collaboration and minimized overlapping efforts. However, we need to refine our task distribution further to optimize efficiency.

One notable area for improvement is our handling of **code integration**. Merging diverging branches proved to be time-consuming and occasionally disrupted our workflow. To address this, we should implement more frequent integration practices and establish clearer guidelines for branch management. This will help us avoid the pitfalls of merging conflicts and enhance our overall productivity.

Looking ahead to the upcoming sprint, we should focus on better organizing the workload across the team. By distributing tasks more evenly and ensuring that everyone is aligned with our coding standards and best practices, we can mitigate potential bottlenecks and maintain a steady progress rate.

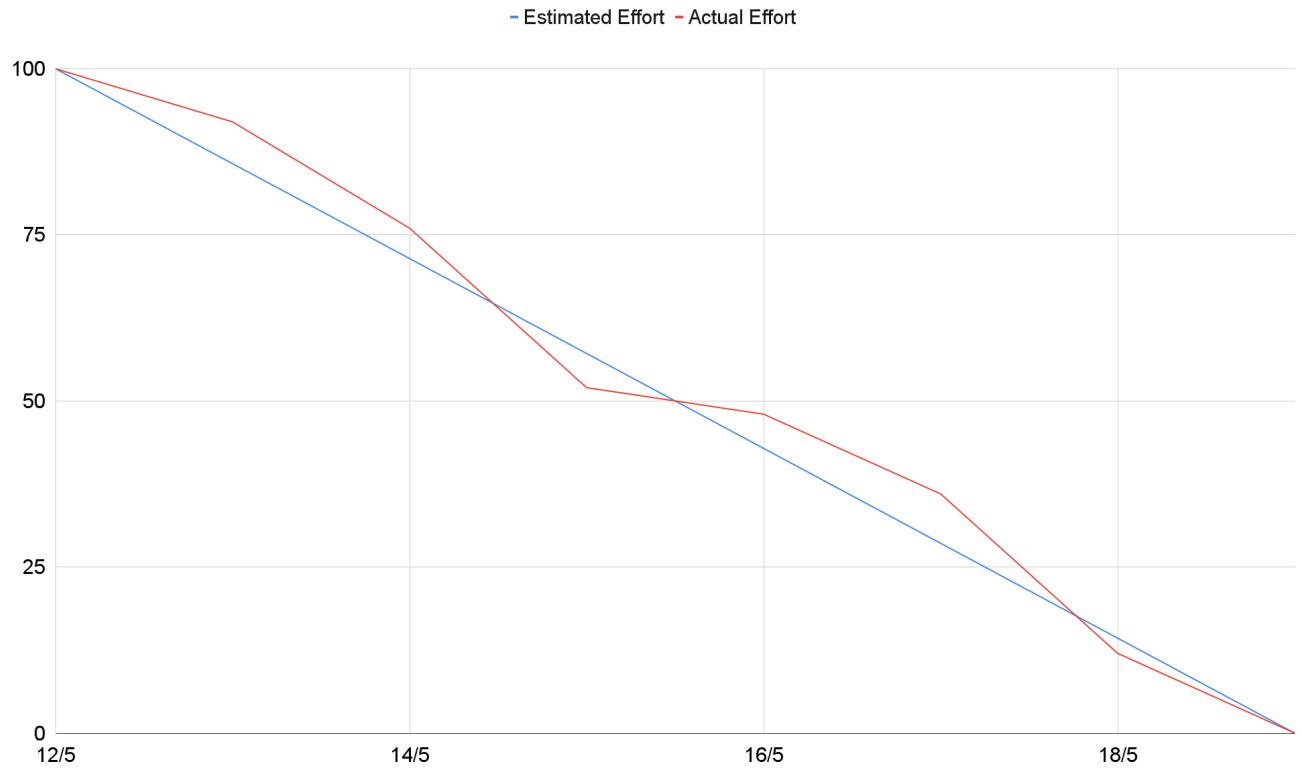
Additionally, we should continue to invest time in learning and mastering the technologies we are using. This will not only boost our efficiency but also increase the quality of our product.

In conclusion, while this sprint has been a success, it also highlighted several areas for potential growth. By addressing these issues and building on our strengths, we can look forward to an even more productive standard for out team as a whole.

5.2.2 Review

Due to the subset of the chosen tasks we did't have too much to show to the "Product Owner" at the end of the sprint. We shown the automatic deployment of the backend on render, and the frontend on GitHub Pages. We also shown the automated tests to the backend functionalities. Overall, the product Owner was satisfied with the results of the sprint, and didn't ask us to change anything.

5.3 Burn-down Chart



6 Sprint 2

6.1 Backlog

					New estimates of effort (remaining at the end of day ...)						
Sprint task	**Product backlog items**	**Volunteer**	**DoD**	**Initial estimate of effort**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Sign up with google	As a user, I want to be able to create an account, so that i am able to access to the mind merge services	Luca S.	The google sign-up and sign-in is implemented	4	2.5	0	0	0	0	0	0
Organization Manager Page	As a manager, since I want to use the platform, I want to be able to create an organization at any moment, so that i can start a new project	Luca S.	The Organization creation is implemented	4	4	4	0	0	0	0	0
	As an owner, as my organization grows, I want to be able to add a user at any time, so that he can start working for my organization		User Management for each Organization is implemented								
External Notification System	As a user, I want to be able to see major updates at a glance with an external notification system (email), so that I get notified when something important happens	Gabriele	Each function for the External Notification System is implemented	4	4	4	2	1.5	0	0	0
JSW token creation	As a user I want to be able to log in to the service with an account i created, so that i am able to access to the mind merge services	Gioele	The log-in functions are correctly implemented	4	4	4	0	0	0	0	0

6.2 Review and Retrospective

6.2.1 Retrospective

Sprint 2 went well overall. However, as reflected in the burndown chart, we underestimated our velocity. This was primarily because we allocated a significant amount of time during the week for another course project that we anticipated would require considerable effort. To our surprise, we completed that project in one day, leaving us with more available time for the rest of the week.

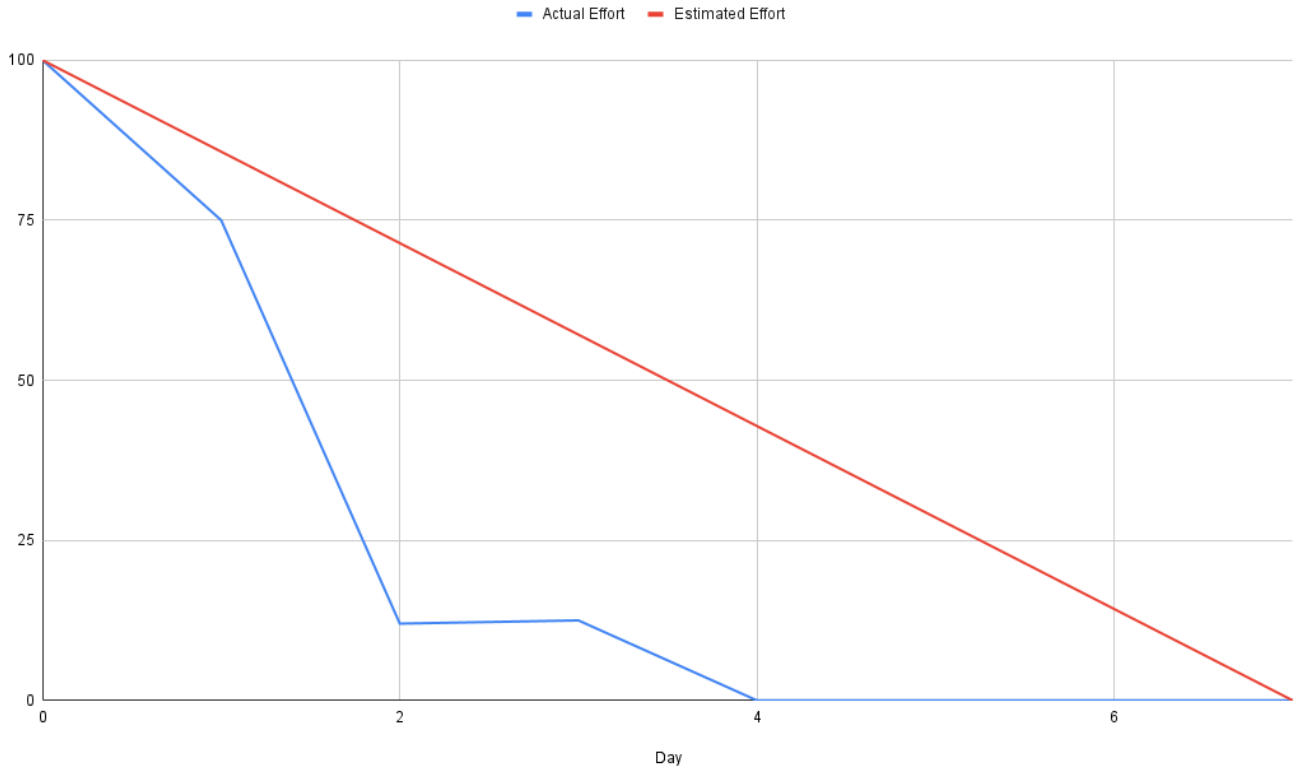
Our main concern going into Sprint 3 is becoming more familiar with the Vue framework. By the end of Sprint 2, we still faced challenges in using it effectively.

6.2.2 Review

We demonstrated the Google sign-in functionality and the "Organization Manager" page to the product owner. This page allows users to create an organization and add or remove users.

The product owner was happy with the results but requested a change. Specifically, he asked that when a new organization is created, it should automatically be selected as the current active organization. We will note this down and address it in one of the upcoming sprints.

6.3 Burn-down Chart



7 Sprint 3

7.1 Backlog

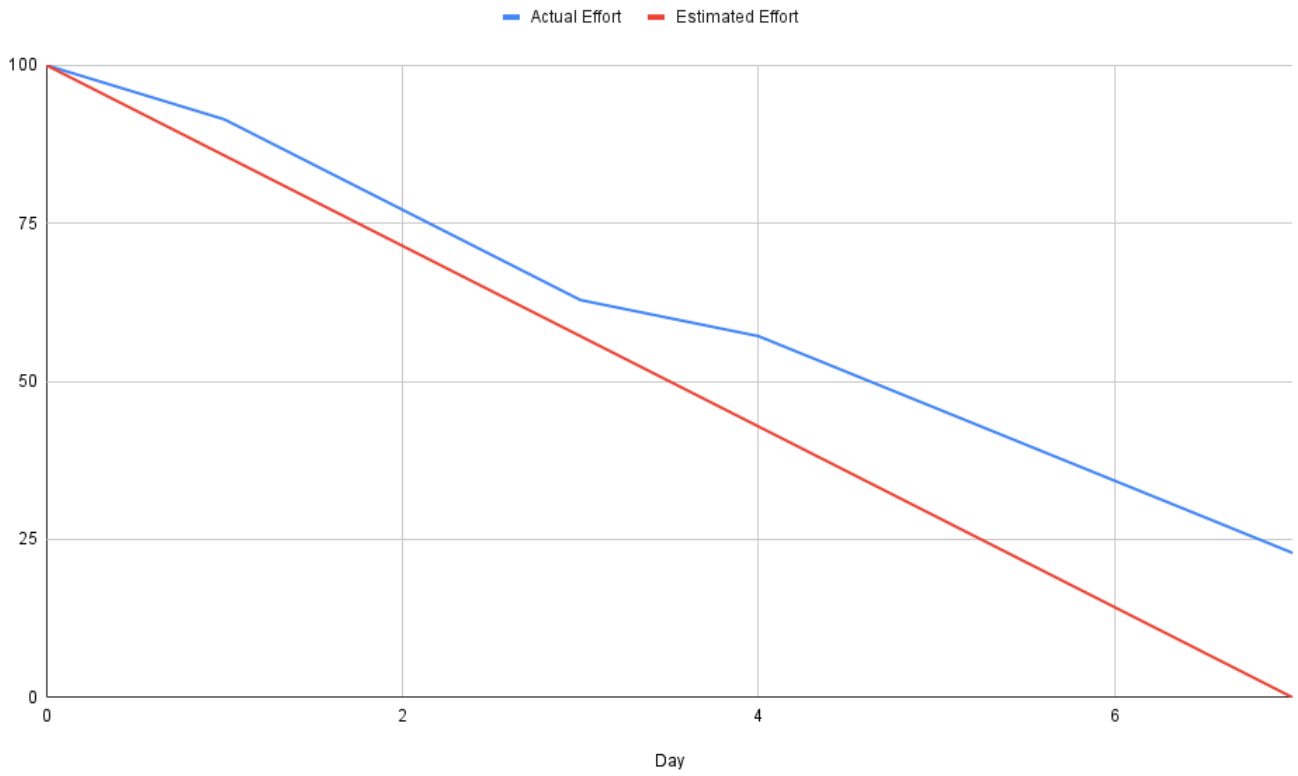
					New estimates of effort (remaining at the end of day)						
Sprint task	**Product backlog items**	**Volunteer**	**DoD**	**Initial estimate of effort**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Task Tree Navigator	As a user, I want to be able to get a visual tree of the tasks and sub-tasks, so that I have a clear idea of what need to be done	Luca S.	Users can easly see taksk in a tree structure and navigate between them	8	5	0	0	0	0	0	0
Create/Delete Task	As a user, I want to be able to freely create a task at any time, so that I can annotate the detail of a project As a user, I want to be able to delete the task at any given time, so that I can keep the documentation updated.	Luca S.	Users alter the task tree by creating or deleting tasks	3	3	3	0	0	0	0	0
Visualiza Selected Task Name	As a user, I want to be able to get a visual tree of the tasks and sub-tasks, so that I have a clear idea of what need to be done	Luca S	The selected task's name is displayed	0.5	0.5	0.5	0	0	0	0	0
Edit Selected Task Name	As a user, I want to be able to edit the task (e.g. title, description etc), so that I can keep the documentation updated.	Luca S	A button is available to edit the selected task's name	0.5	0.5	0.5	0	0	0	0	0
Visualiza Selected Task Notes	As a user, I want to be able to get a visual tree of the tasks and sub-tasks, so that I have a clear idea of what need to be done	Gabriele	It is possible to visualize all the notes that have been written for a specific task	6	6	6	4	2	2	0	0
Edit Selected Task Notes	As a user, I want to be able to edit the task (e.g. title, description etc), so that I can keep the documentation updated.	Gabriele	It is possible to edit one of the notes that have been written for a specific task	6	6	6	6	6	2	0	0
Visualiza Selected Task Description	As a user, I want to be able to get a visual tree of the tasks and sub-tasks, so that I have a clear idea of what need to be done	Luca F	It is possible to visualize the description of the selected task	4	4	4	4	4	4	4	4
Edit Selected Task Description	As a user, I want to be able to edit the task (e.g. title, description etc), so that I can keep the documentation updated.	Luca F	It is possible to edit the description of the selected task	4	4	4	4	4	4	4	4
Log In with google	As a user I want to be able to log in to the service with an account i created, so that i am able to access to the mind merge services	Gioele	After a sign up, it is possible to log in from the "log in" page of the mindmerge website	4	4	4	4	4	4	4	0

7.2 Review and Retrospective

7.2.1 Retrospective

7.2.2 Review

7.3 Burn-down Chart



8 Api Design

9 Testing Implementation

10 How to demo

In this section We'll explain everything needed to demo and evaluate our project.

10.1 Demo Accounts

To log in in the MindMerge account you'll need to use a Google account, that has been inserted in Google's developer console as a testing account. This is because our app has not been published and verified by google, therefore there are some limitations on the accounts that can log in.

We have created two testing account, and we sent the credentials to you via email. The email has been sent Monday 24th of June at 8:00 AM, with the subject "MindMerge - Testing Account".

If you prefer to use you own account, you can send us the email address you want to use, and we'll add it to the list of testing accounts.

10.2 Demo Limitations

Our demo has some minors limitations that stands from the fact that we are using a free tier of basically every service we are using. We would like to quickly list them here so that you are aware of them before starting the demo.

- **LLM Service:** We are using the free tear of the Google Bard API as our llm service. The free tear is limited to 15 requests per minute. One report generation uses as many requests as there are tasks in the task tree... This means that in one minute you can generate at most:

  - 15 reports for a task that has no child tasks
  - 5 reports for a task that has 3 child tasks
  - 1 report for a task that has 15 child tasks

  and so on...

- **Backend startup time:** The backend is hosted on render, with the free tear... this means that our service will be shut down if it is not used for a while. According to the documentation this can "delay requests by 50 seconds or more".

  To quickly check whether the backend is up and running, you can go to the following URL: https://mindmerge-backend.onrender.com/hello

- **Backend response time:** As mentioned early we used the free google bard API, however the free tear is not available in the EU, therefore we had to position the the backend in the US, this makes the interface a bit slow, as every request made to the backend has to travel across the pacific ocean twice.

- **Notification email:** Our system will send you an email with a report every time you request it... however, we are using the free tear of the services that allow us to send emails, therefore the emails will be sent from a random email address, and this makes it very likely that **the email will be marked as spam**.

10.3 Video

10.4 Instructions