

## Developer’s Projects Manager & Tasks Tracker

Senior Project By

**Ali Fakih**

### Submitted to the School of Arts & Science of the Lebanese International University

In part of fulfillment of the requirements for the degree of

**BACHERLOR OF SCIENCE IN**

**COMPUTER SCIENCE**

**Supervised by: Dr. Ali Choumane**

### Spring 2023-2024

Dedication:

First of all, I am going to dedicate this project to our almighty God, who gave us strength and knowledge for our everyday life.

To my beloved parents and friends, for always supporting me in our studying and work, and for helping me find and realize my potential.

Last to my respected professor Ali Choumane for leading me step by step to reach my aim and to overcome every single obstacle I faced during the time I worked on my project and even before that during my three years of studying.

**Acknowledgment:**

I have really worked hard and put so many efforts in my "Developer’s Projects Manager & Tasks Tracker" project. However, it would neither have been completed on time nor looked like I wanted without the kind and special support of some individuals. I would like to extend my sincere thanks to all of them.

I am highly indebted to my professor Ali Choumane for his guidance and constant supervision as well as for providing necessary information regarding my project.

My thanks and appreciations also go to my parents and friends for their encouragement that helps me finish my project perfectly and on time.

**Abstract:**

I am creating an online ”Office Automation System” that allows both developers and leaders to manage and track their work in a company.

The system presents so many features for both developers and managers. To explain more, the most important features for leaders are to view, add and archive projects and tasks, as well as tracking the progress of developers on each task. As for developers, they can view their assigned tasks and check their current states and work time. Moreover, developers can start new tasks as well as pause/resume work on certain tasks. When done, they can choose to mark the task as “done” for the company leader to see it later on their side.

Throughout my proposal, I am going to explain every single feature available in my application in details.

To sum up, the purpose of this system is to provide a special service for companies so that they can enjoy flexible remote work from anywhere and at any time.

# List of Tables:

[Table 1- Search Projects use case 18](#Table1_SearchProejctsUsecase)

[Table 2 – Edit Project use case 19](#Table2_EditProjectUsecase)

[Table 3 – Add Task use case 20](#Table3_addTaskUsecase)

[Table 4 – Set as “completed” use case 21](#Table4_setAsCompletedUsecase)

[Table 5 – Create Developer use case 22](#Table5_createDeveloperUsecase)

[Table 6 – Archive Developer use case 23](#Table6_archiveDeveloperUsecase)

[Table 7 – Manage Tasks use case 24](#Table7_manageTasksUsecase)

[Table 8 – Start New Task use case 25](#Table8_startNewTaskUsecase)

[Table 9 – Statistics use case 26](#Table9_viewStatisticsUsecase)

List of Figures:

[Figure 1- Use Case Diagram 17](#Figure1_usecaseDiagram)

[Figure 2 - ER-Diagram 35](#Figure2_ERDiagram)

[Figure 3 - Leader’s Main Page 36](#Figure3_leaderMain)

[Figure 4 – Leader’s Projects Page 38](#Figure4_leadersProjects)

[Figure 5- Leader’s View Project Page 38](#Figure5_leadersViewProject)

[Figure 6 – Edit Project 39](#Figure6_editProject)

[Figure 7 – Leader’s Tasks 39](#Figure7_leadersTasks)

[Figure 8 – Leader’s Tasks’ Icons 40](#Figure8_leadersTasksIcons)

[Figure 9 – Leader’s View Task 41](#Figure9_leadersViewTask)

[Figure 10 – Edit Task Page 41](#Figure10_leadersEditTask)

[Figure 11 – Add Task 42](#Figure11_addTask)

[Figure 12 – Leader’s View Developers 42](#Figure12_leadersViewDevelopers)

[Figure 13 – View developer’s Tasks 42](#Figure13_viewDevelopersTasks)

[Figure 14 – Add Developer 43](#Figure14_addDeveloper)

[Figure 15 – Developer’s Main Page 43](#Figure15_developersMain)

[Figure 16 – Developer’s Tasks 44](#Figure16_developersTasks)

[Figure 17 – Developer’s Statistics Page 44](#Figure17_developersStatistics)

Table of Contents:

Contents

Dedication… II

Acknowledgment III

Abstract IV

List of Tables V

List of Images VI

[Chapter 1 9](#_bookmark0)

[1.1-Introduction: 9](#_bookmark1)

[1.2-Objective: 9](#_bookmark2)

[1.3-Scope: 10](#_bookmark3)

[1.4-Technology Constraints 10](#_bookmark4)

[Software needed: Visual studio code, Xampp, Web browser 10](#_bookmark5)

[1.5-Problem 11](#_bookmark6)

[1.6-Solution 11](#_bookmark7)

[Chapter 2 12](#_bookmark8)

[2.1-Functional and Conceptual Study 12](#_bookmark9)

[2.1.1-Functional Requirements 12](#_bookmark10)

[2.1.2-Nonfunctional requirements 15](#_bookmark11)

[2.2-UML Use Case Diagram & Scenario 16](#_bookmark16)

[2.2.1-Use Case Diagram 16](#_bookmark17)

[2.2.2-Use Case scenarios (Leader actor) 18](#_bookmark19)

[2.2.3-Use Case scenarios (Developer actor) 30](#UseCaseScenarios_Developer)

[Chapter 3 34](#_bookmark53)

[3.1-System Design: 34](#_bookmark54)

[3.1.1-ER-Diagram 34](#_bookmark55)

[ER Diagram Figure: 35](#Figure2_ERDiagram)

[Chapter 4 37](#_bookmark59)

[Implementation: 37](#_bookmark60)

[4.1-Introduction: 37](#_bookmark61)

[4.2- Project’s Parts: 38](#_bookmark62)

[4.2.1- Leader’s Main Page: 38](#_bookmark63)

[4.2.2- Leader’s Projects Page: 41](#_4.2.2-_Leader’s_Projects)

[4.2.3- View Project Page: 42](#_4.2.3-_View_Project)

[4.2.4- Edit Project Page: 43](#_4.2.4-_Edit_Project)

[4.2.5- Leader’s Tasks Page: 44](#_4.2.5-_Leader’s_Tasks)

[4.2.6- Leader’s View Task Page 45](#_4.2.6-_Leader’s_View)

[4.2.7- Edit Task Page: 47](#_4.2.7-_Edit_Task)

[4.2.8- Add Task Page: 49](#_4.2.8-_Add_Task)

[4.2.9- Leader’s View Developers Page: 50](#_4.2.9-_Leader’s_View)

[4.2.10- View Developer’s Tasks: 51](#_4.2.10-_View_Developer’s)

[4.2.11- Add Developer Page: 52](#_4.2.11-_Add_Developer)

[4.2.12- Developer’s Main Page: 53](#_4.2.12-_Developer’s_Main)

[4.2.13- Developer’s Tasks Page: 54](#_4.2.13-_Developer’s_Tasks)

[4.2.14- Developer’s View Statistics Page: 55](#_4.2.14-_Developer_View)

[Chapter 5 70](#_bookmark136)

* 1. [Conclusion: 70](#_bookmark137)
  2. [Future Work 70](#_bookmark138)

[References 71](#_bookmark139)

### Chapter 1

In this chapter, I am going to state the problems that I found, and how my project helps in solving them, in addition to mentioning technology constraints and my objectives, then explaining all of them in details.

### 1.1-Introduction:

Now a days, Computer Science has become a very popular major across the world. Almost any company requires employees of this major to make use of relative technologies that matches their style and needs. With Computer Science employees in companies, leaders are to keep manage project scale tasks distribution. “Developer’s Projects Manager & Tasks Tracker” makes this job easier for both leaders and developers in companies across the world. I am going to discuss the advantages later, but to end. For many reasons, Computer Science based companies are becoming increasingly popular across the world.

### 1.2-Objective:

The main Objective of this platform (website) for my project is to get rid of all difficulties that can be faced in physical task management, to reach wider audience and to cancel the limitation of space, time and locations, beside increasing user satisfaction of course.

### 1.3-Scope:

This section describes the requirement analysis of the system; it aims to get rid of what users face in traditional task management. This system can be implemented to any Computer-Science-based company of which their work includes distribution of tasks of projects with tracking needs.

### 1.4-Technology Constraints:

**Software needed:** Visual studio code, Xampp, Web browser

**Languages:** HTML, CSS, JavaScript, PHP, SQL **Frameworks & libraries:** JQuery. **Hardware:** Computer.

### 1.5-Problem:

In traditional companies, stakeholders face many problems:

* Company leaders have to contact their developers when they want to check on their progress.
* Developers needs to contact their leader every time they finish a task
* Company leaders have to ask their developers about their working time.
* Communication is restricted to daytime in most companies.

### 1.6-Solution:

Creating an online office automation system that allows the following:

* Company leaders can reach check their developer’s and their progress anytime without contacting them.
* Developers can simply mark a task as “done”, and the leader can see it anytime.
* Company leaders can check work time on their developers or tasks anytime without making any contacts.
* The system us usable 24/7 since it doesn’t require both ends to be online at the same time.
* Unique user experience.

### Chapter 2

This chapter describes both functional and non-functional requirements for my project. In addition to that it shows the UML use cases and scenarios with all related details.

2.1-Functional and Conceptual Study

#### 2.1.1-Functional Requirements:

1. **Sign up:** Thesystem allows guests to create a new account to access the site.
2. **Login:** The system allows guests to login easily to continue managing or working on their tasks using their email and password.
3. **View Projects:** The system allows the leader to view their own projects
4. **Search Projects:** The system allows the leader to search for a project.
5. **Create Project:** The system allows the leader to create a new project.
6. **Edit Project:** The leader shall be able to edit his own projects. Changing their names, and adding tasks to them.
7. **Archive Project:** The leader shall be able to archive his own projects.
8. **View Tasks:** The leader shall be able to view his own tasks.
9. **Filter Tasks:** The leader shall be able to filter his own tasks for more precise seeking.
10. **View Task:** The leader shall be able to view additional details about a single task.
11. **Edit Task:** The leader shall be able to edit a task of his own creation.
12. **Mark Task as Completed:** The leader shall be able to a task as “completed”.
13. **Return Task:** The leader shall be able to return a “done” task to the developer marking it as “on hold”.
14. **Archive Task:** The leader shall be able to archive any task of his creation.
15. **View Developers:** The leader shall be able to view his own Developers.
16. **Search Developers:** the leader shall be able to search for a developer by name or email allowing precise seeking.
17. **Add New Developer:** The leader shall be able to add a create a new developer account.
18. **View Developer:** The leader shall be able to view additional details about a developer.
19. **Archive Developer:** The leader shall be able to archive a developer.
20. **View Tasks:** A developer shall be able to view their own assigned tasks.
21. **Start Task:** A developer shall be able to start a newly assigned task,
22. **Mark as in progress:** A developer shall be able to mark a task that is “on hold” as “in progress”.
23. **Mark as on hold:** a developer shall be able to mark as task that is “in progress” as “on hold”.
24. **Mark as done:** a developer shall be able to mark a task that is either “in progress” or “on hold” as “done”.
25. **View Statistics:** a developer shall be able to see their own statistics in a company.
26. **Un/Collapse Project Statistics:** a developer shall be able to collapse or un-collapse a project’s statistics.

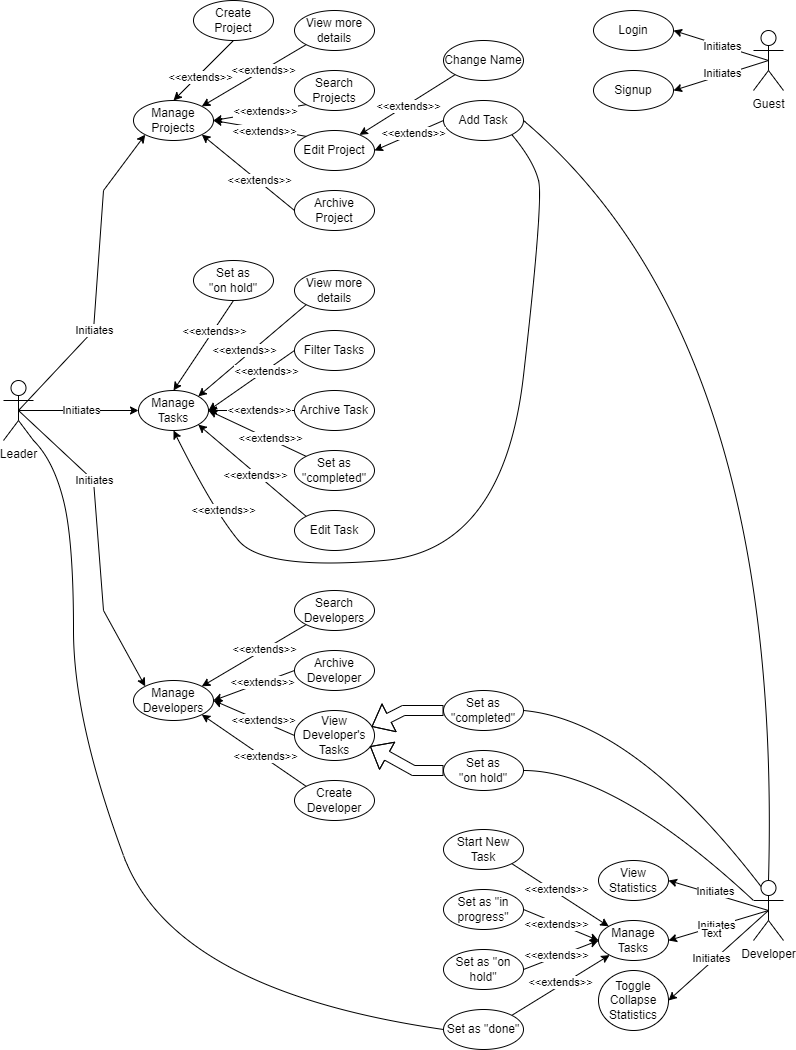
#### 2.1.2-Nonfunctional requirements:

1. **Login:**  Username and password must both be valid for the same user for the system to allow the guest to continue as that user.
2. **Signup:** ‘Name’ must be alphabetic or spaces only and can’t already exist in the database. It will ignore spaces at the start, spaces at the end, as well as 2 consecutive spaces in the middle. ‘Email’ must be a valid format. The system ignores spaces at the start as well as at the end. Email shouldn’t already exist for users. ‘Password’ and ‘Confirm Password’ must be identical and must be at least 8 characters long.
3. **Create Project:** The system only allows project names that contains alphanumerical characters or spaces. It ignores spaces at the start, spaces at the end, and 2 consecutive spaces in the middle. The project name should be unique for the current leader.
4. **Edit Project:** The system only allows project names that contains alphanumerical characters or spaces. It ignores spaces at the start, spaces at the end, and 2 consecutive spaces in the middle. The project name should be unique for the current leader.
5. **Create Task & Edit Task:** A task should have a unique name for the current project. Estimated hours can’t exceed 999 while Estimated minutes and Estimated seconds can’t exceed 99.
6. **Filter Tasks:** A filter option must be active before the system can filter by it.
7. **Add New Developer:** All data except “phone” are required. “First Name” & “Last Name” must be alphabetical characters or spaces only, and the system ignores any spaces at the start, any spaces at the end, as well as any 2 consecutive spaces in the middle. “Email” must be a valid one. The system ignores spaces at the start and at the end of “Email”. “Phone” (if given) can only contain the following characters: “/”, “-”, spaces and numerical characters. Spaces at the start and at the end are ignores in a phone number. ”Hours Per Day” must be contain numerical characters only and must be between 1 and 24. Any spaces at the start of at the end are ignored.

#### 2.2-UML Use Case Diagram & Scenario

2.2.1-Use Case Diagram

Leader [actor](https://www.uml-diagrams.org/use-case-actor.html) uses the website to make manage and track the company’s tasks while developers use it to track their own tasks and statistics. Any user that isn’t logged in is named a Guest. (See [figure 1](#Figure1_usecaseDiagram))



*Figu**re 1- Use Case Diagram*

2.2.2-Use Case scenarios (leader actor)

In this part we will be explaining the use cases of the leader in details, regarding primary actors, main flow, and the alternative one. In addition to a short description for each use case making all points clear.

*Login use case*

Check “[Table 1](#Table1_SearchProejctsUsecase)” scenario to see related details regarding search projects use case.

*Table 1- Search Projects use case.*

|  |
| --- |
| **Use Case Name:** Search Projects |
| **Actors:** Leader |
| **Brief Description:** This use case allows the leader to search for given projects. |
| **Basic Flow:**   1. Leader types some text in a text field. 2. System fetches all projects of the leader’s creation that contains the text given by the leader. 3. The system displays the results for the leader. |
| **Alternative Flow:**   * **No results were found:** the system displays a simple message that says no projects were found with the given text. |
| **Pre-conditions:**   * Leader must have internet connection. |
| **Post-conditions:**   * In case the system found a result, it should display the results. * In case the system didn’t find any results, then it displays an empty table. |

*Edit Project use case*

For more details about Edit Project use case, see “[Table 2](#Table2_EditProjectUsecase)” scenario

*Table 2 – Edit Project use case*

|  |
| --- |
| **Use Case Name:** Edit Project |
| **Actors:** Leader |
| **Brief Description:** Navigates the leader to the “Edit Project” page. |
| **Basic Flow:**   1. Leader clicks on the Edit icon on a project. 2. System takes the leader to the “Edit Project” page. |
| **Alternative Flow:**  None. |
| **Pre-conditions:**  None. |
| **Post-conditions:**   * System should redirect the user to the “Edit Project” page. |

*Add Task use case*

See “[Table 3](#Table3_addTaskUsecase)” scenario, to check all details related to this case from the actors, to main and alternative flow, and so on.

*Table 3 – Add Task use case*

|  |
| --- |
| **Use Case Name:** Add Task |
| **Actors:** Leader |
| **Brief Description:** This use case allows the leader to add a new task for a specific project. |
| **Basic Flow:**   1. Leader fills the information for the new task and clicks add. 2. System adds the new task and sends it to the assigned developer. |
| **Alternative Flow:**   * **Input information were invalid**: the system should display an error message informing the leader of the exact problem. * **There was a problem connecting to the server:** the system should display an error. |
| **Pre-conditions:**   * Leader must input data correctly and have internet access. |
| **Post-conditions:**   * In case the input data were correct and connection was success, the system should add the new task and send it to the assigned developer. * In case the input data was invalid: System should display an error message showing the exact problem. * In case connection was failed: system should display an error message. |

*Set as “completed” use case*

For more details regarding Set as “completed” use case, see “[Table 4](#Table4_setAsCompletedUsecase)” case scenario.

*Table 4 – Set as “completed” use case*

|  |
| --- |
| **Use Case Name:** Set as “completed” |
| **Actors:** Leader |
| **Brief Description:** This use case allows the leader to set the current task as “completed”. |
| **Basic Flow:**   1. Leader clicks the “set as completed” icon. 2. System sets the task as “completed” and confirms for the leader. |
| **Alternative Flow:**   * **No connection:** System displays an error message. |
| **Pre-conditions:**   * Leader must have Internet access. |
| **Post-conditions:**   * In case connection was successful, the system should set the task as “done” and confirm for the leader. * In case of connection issues, the system should display an error message. |

*Create Developer use case*

See “[Table 5](#Table5_createDeveloperUsecase)” scenario to check the description of the use case and its flow.

*Table 5 – create developer use case*

|  |
| --- |
| **Use Case Name:** Create Developer |
| **Actors:** Leader |
| **Brief Description:** This use case allows the leader to add a new developer account. |
| **Basic Flow:**   1. Leader fills the required information for the new account and clicks “Create”. 2. System checks the input data and creates the new account then confirms. |
| **Alternative Flow:**   * **Input invalid:** System displays an error message showing the exact invalidation. * **No connection:** System should display an error message. |
| **Pre-conditions:**   * Leader should enter correct input and have internet access. |
| **Post-conditions:**   * In case the input was valid and connection was established, the system should create the new developer account and confirm for the leader. * In case the input data were invalid, the system should display an error message showing the exact invalidation. * In case the connection failed, the system should display an error message. |

*Archive Developer use case*

See “[Table 6](#Table6_archiveDeveloperUsecase)” scenario to check information related to primary actor, in addition to brief description of view products use case and its flow.

*Table 6 – archive developer use case*

|  |
| --- |
| **Use Case Name:** Archive developer. |
| **Actors:** Leader |
| **Brief Description:** This use case allows the leader to archive a developer account. |
| **Basic Flow:**   1. Leader clicks the archive icon for an existing developer. 2. System confirms that the developer’s account was successfully archived. |
| **Alternative Flow:**   * **No connection:** the system displays an error message. |
| **Pre-conditions:**   * Leader must have internet access. |
| **Post-conditions:**   * In case connection was successful, the system should archive the developer’s account. * In case connection was not successful, the system should display an error message. |

*Manage Tasks use case*

See “[Table 7](#Table7_manageTasksUsecase)” scenario to check information related to primary actor, in addition to brief description of make order use case and its flow.

*Table 7 – manage tasks use case*

|  |
| --- |
| **Use Case Name:** Manage Tasks |
| **Actors:** Leader |
| **Brief Description:** This use case navigates the leader to the “Tasks” page. |
| **Basic Flow:**   1. Leader clicks on “Manage Tasks” 2. System navigates the leader to the “Tasks” page. |
| **Alternative Flow:**   * None. |
| **Pre-conditions:**   * None. |
| **Post-conditions:**   * System should navigate the leader to the “Tasks” page. |

2.2.3-Use Case scenarios (Developer actor)

In this part I will be explaining the use cases of the developer in details, regarding primary actors, main flow, and the alternative one. In addition to a short description for each use case making all points clear.

*Start New Task use cases*

Check “[Table 8](#Table8_startNewTaskUsecase)” use case scenario to view all details related to this use case, from its flow, to the primary actors, extended use cases and included one, in addition to a brief explanation regarding it.

*Table 8 – Start new task*

|  |
| --- |
| **Use Case Name:** Start New Task |
| **Actors:** Developer |
| **Brief Description:** This use case allows the developer to mark a new task given by a leader as “in progress” for the first time. |
| **Basic Flow:**   1. The developer clicks the “start” icon. 2. The System changes the status of the task as “in progress”. |
| **Alternative Flow:**   * **There was no connection to the server:** the system displays an error message for the developer. |
| **Pre-conditions:**   * Developer must have internet connection. |
| **Post-conditions:**   * In case there was no connection, the system will automatically display an error message for the developer. |

*View Statistics use case*

Check “[Table 9](#Table9_viewStatisticsUsecase)” scenario to view all details related to this use case, from its flow, to the primary actors, extended use cases and included one, in addition to a brief explanation regarding it.

*Table 9 - View statistics*

|  |
| --- |
| **Use Case Name:** View Statistics |
| **Actors:** Developer |
| **Brief Description:** This use case allows the developer to view statistics related to his work history. |
| **Basic Flow:**   1. User clicks view statistics. 2. System redirects the developer to the “Statistics” page. |
| **Alternative Flow:**   * None. |
| **Pre-conditions:**   * None |
| **Post-conditions:**   * System should redirect the user to the “Statistics” page. |

### Chapter 3

This chapter describes the system design, including the data base of the project, the related tables and the relations between them.

### 3.1-System Design:

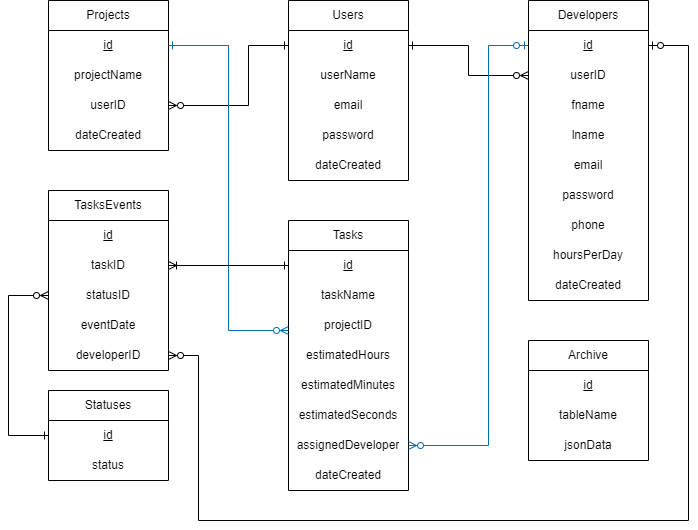
#### 3.1.1-ER-Diagram

This ER (Entity Relationship) Diagram represents the model of the system. It shows all the instruments of database tables and the relations between them. The main entities of this system are: Archives, Developers, Projects, Statuses, Tasks, TaskEvents, and Users. (See [figure 2](#Figure2_ERDiagram))

Here are the entities and their attributes:

* Archives: id (primary key), tableName, jsonData.
* Developers: id (primary key), userID (Foreign key), fname, lname, email, password, phone, hoursPerDay, dateCreated.
* Projects: id (primary key), projectName, userID (foreign key), dateCreated.
* Statuses: id (primary key), status.
* Tasks: id (primary key), taskName, projectID (foreign key), estimatedHours, estimatedMinutes, estimatedSeconds, assignedDeveloper (foreign key), dateCreated.
* TasksEvents: id (primary key), taskID (foreign key), statusID (foreign key), eventDate, developerID (foreign key).
* Users: id (primary key), userName, email, password, dateCreated.

ER Diagram Figure:



*Figure 2 - ER-Diagram*

### Chapter 4

This chapter describes the whole view of my website, it shows the design of the application in addition to stating the main role of each page or button.

### Implementation:

#### 4.1-Introduction:

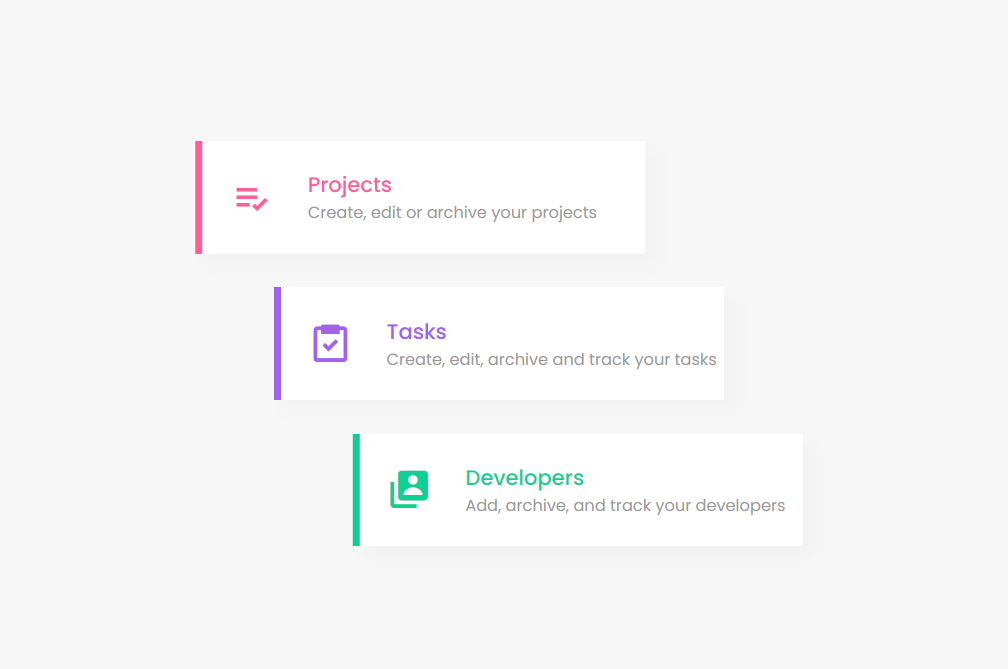
The implementation of this system simply shows how the interface of will look like and how it is applied in order to ensure good performance for the users. The main aim is to create a simple interface that is almost suitable for all users, I tried my best to make it as simple as possible and comfortable for the eyes. It is enough for the user to enter the website and then he/she can easily flow from page to page without any difficulty or interruptions.

Below we are going to view different pages of my website and how they are going to be in details.

4.2- Project’s Parts:

##### 4.2.1- Leader Main Page:

The first page that a leader sees when logging in is the leader’s main page. It contains a list of options to navigate through different pages. Every item in that list is a clickable link that navigates the leader to a different page accordingly. (See [figure 3](#Figure3_leaderMain))



*Figure 3 – Leader’s main page.*

##### 4.2.2- Leader’s Projects Page:

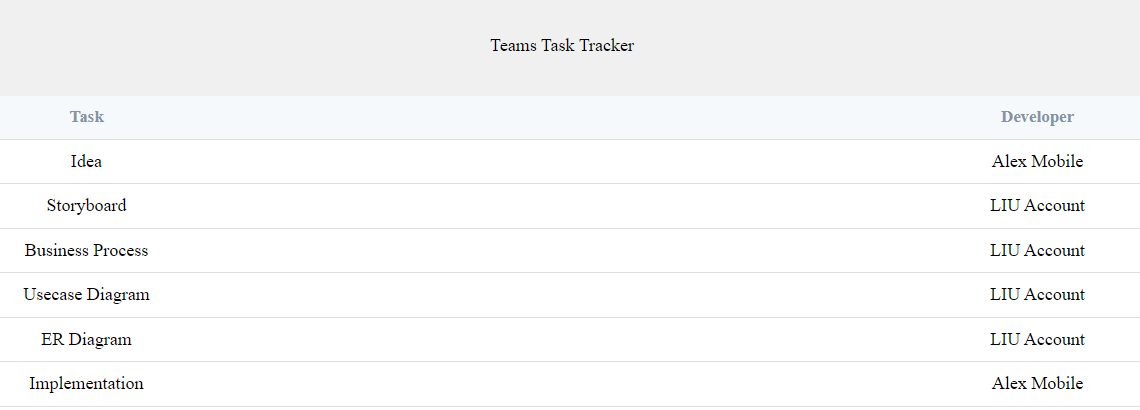
[Figure 4](#Figure4_leadersProjects) show the projects page which displays all the projects the current leader has along with some information about it. From here, the leader may click on the show icon beside a project to see more details about a project (see [figure 5](#Figure5_leadersViewProject)), the edit icon to edit project name or add new tasks to it (See [figure 6](#Figure6_editProject)), and the archive icon to archive the project. The leader may also search for a specific project by its name in the search text field and create a new empty project by clicking the Create Project button with the name typed in the text field.



*Figure 4- Leader’s projects page*

##### 4.2.3- View Project Page:

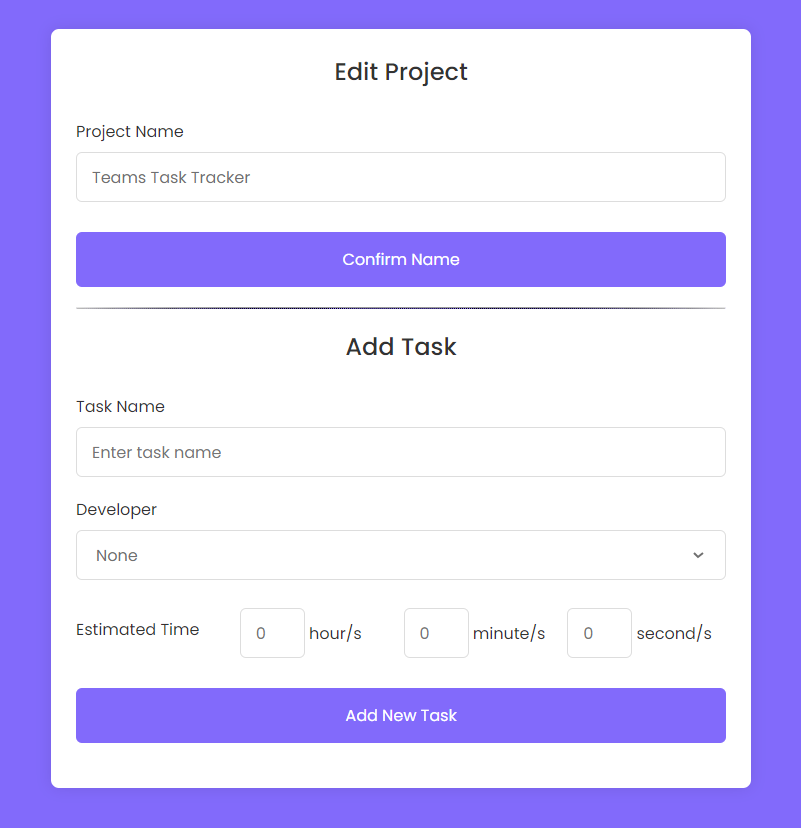
This page will display all the task of a given project along with their assigned developers at the current time. Allowing the leader to easily know who is working on a project and on which tasks specifically. (See [figure 5](#Figure5_leadersViewProject))



*Figure 5- Leader’s view project page*

##### 4.2.4- Edit Project Page:

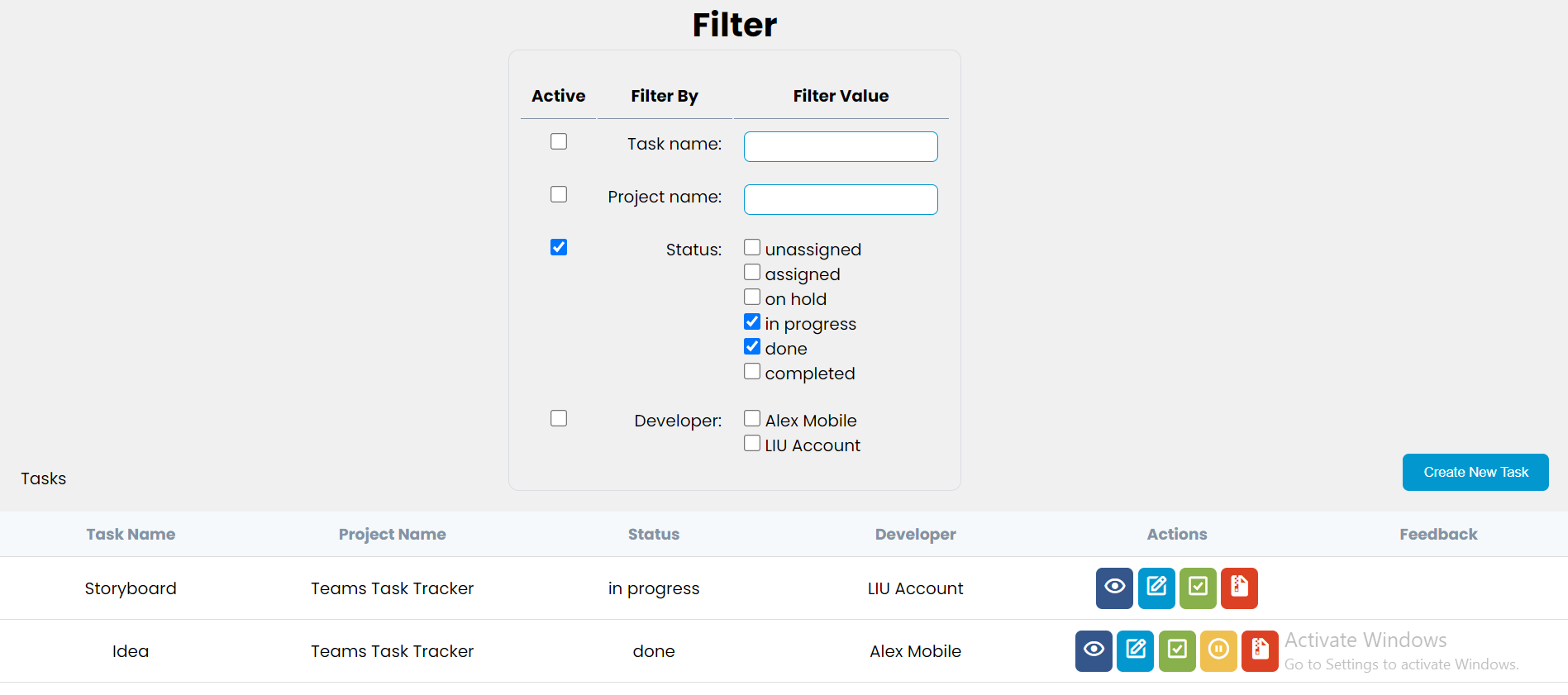
In this page the leader can change the name of a given project as well as add new tasks to it. (See [figure 6](#Figure6_editProject))



*Figure 6 – Edit Project*

##### 4.2.5- Leader’s Tasks Page:

This page displays all the existing tasks for the current leader. Additionally, the leader may filter the results for more precise seeking. For example, the leader may activate the “status” filter by checking the checkbox for it, and then choose to seek for tasks which are either “in progress” or “done”. The leader may also click the Create New Task button to create a new task in a different web page. (See [figure 7](#Figure7_leadersTasks))



*Figure 7 – Leader’s Tasks*

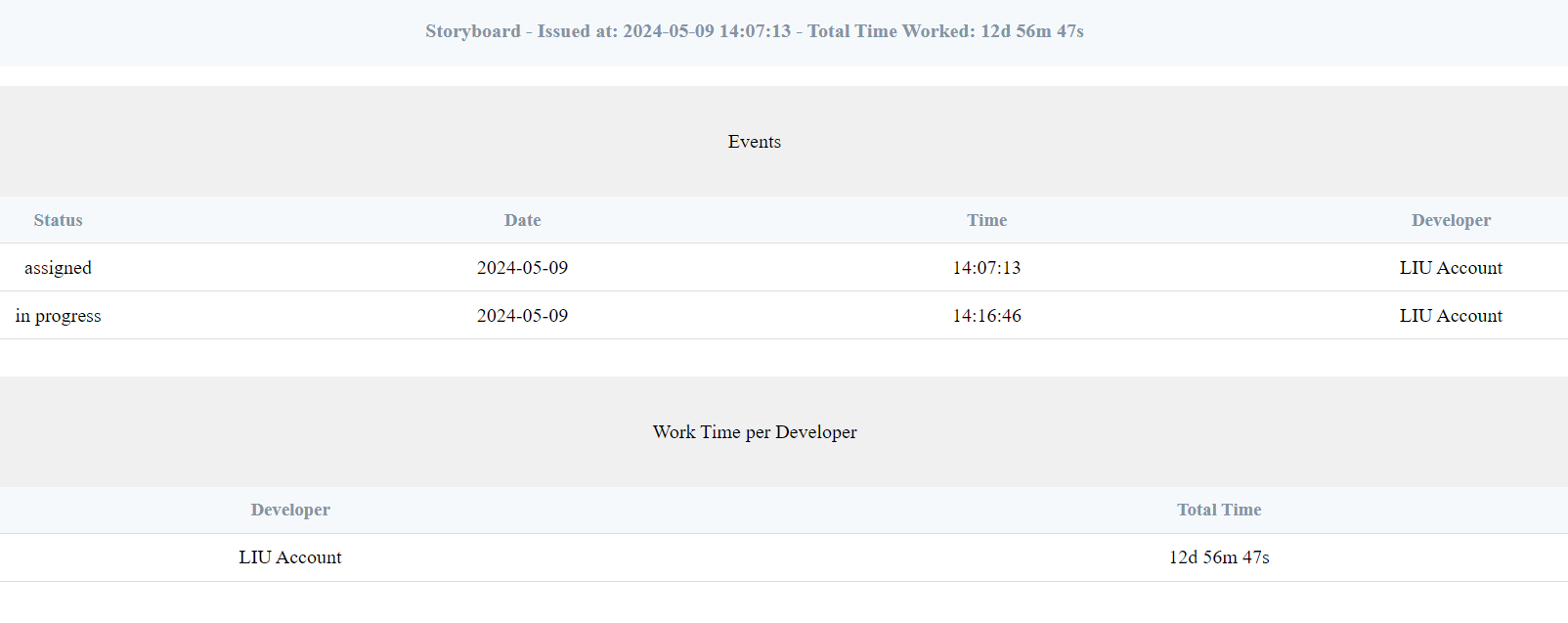
[Figure 8](#Figure8_leadersTasksIcons) shows how the following icons can be clicked to perform different actions. To start with, the show button navigates to a different web page which shows more details about a task (see [figure 9](#Figure9_leadersViewTask)). The edit icon allows the user to edit a task in a different web page (see [figure 10](#Figure10_leadersEditTask)). The Check icon marks the current task as “completed”. The pause icon marks the current task as on hold, which in this context means returning the task to its developer as unfinished with the “on hold” status. The archive icon archives the task.



*Figure 8 – Leader’s Tasks’ Icons*

##### 4.2.6- Leader’s View Task Page:

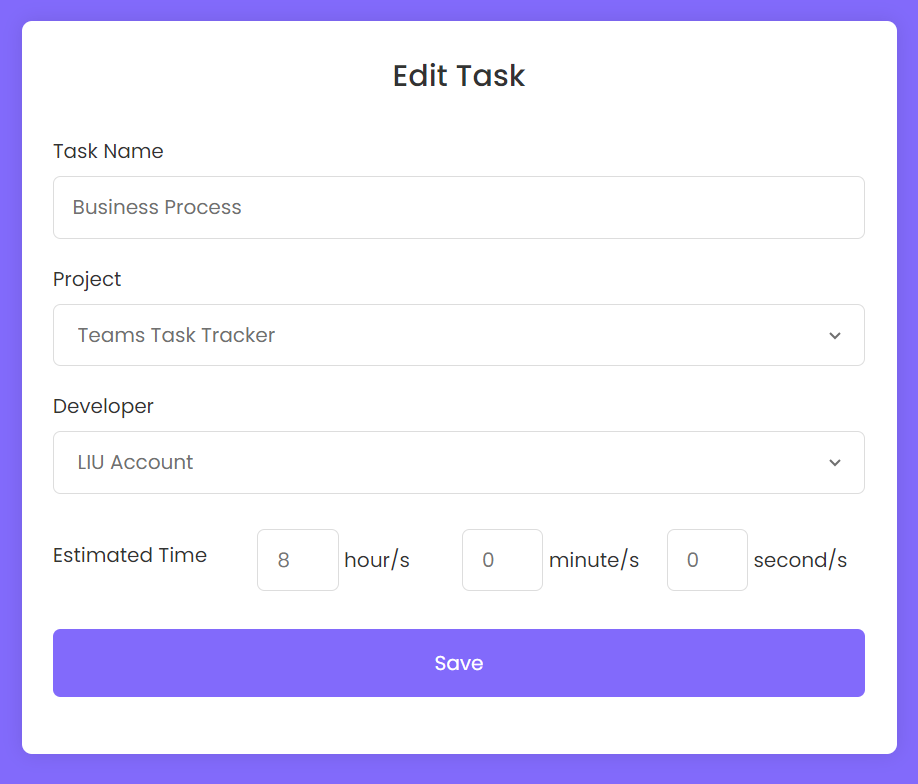
This page shows the leader more details about a task. Details like creation date and total time spent on the current task. Moreover, the time stamps it went through since it was created. Finally, the total time every developer has worked on the task till the current time stamp. (See [figure 9](#Figure9_leadersViewTask))



*Figure 9 – Leader’s View Task*

##### 4.2.7- Edit Task Page:

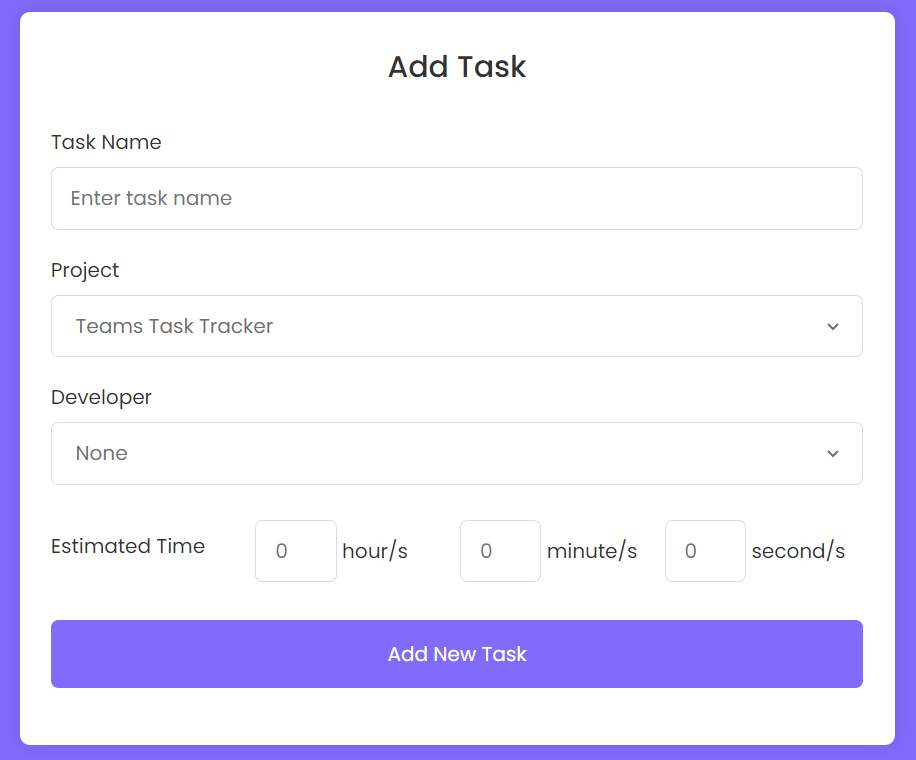
In this page the leader can edit any task. (See [figure 10](#Figure10_leadersEditTask))



*Figure 10 – Edit Task Page*

##### 4.2.8- Add Task Page:

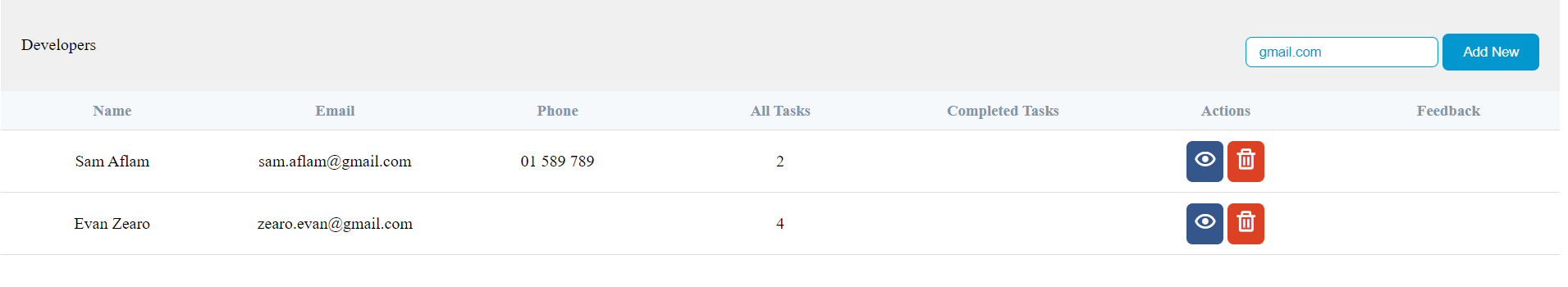
In this page the leader can add a new task. (See [figure 11](#Figure11_addTask))



*Figure 11 – Add Task*

##### 4.2.9- Leader’s View Developers Page:

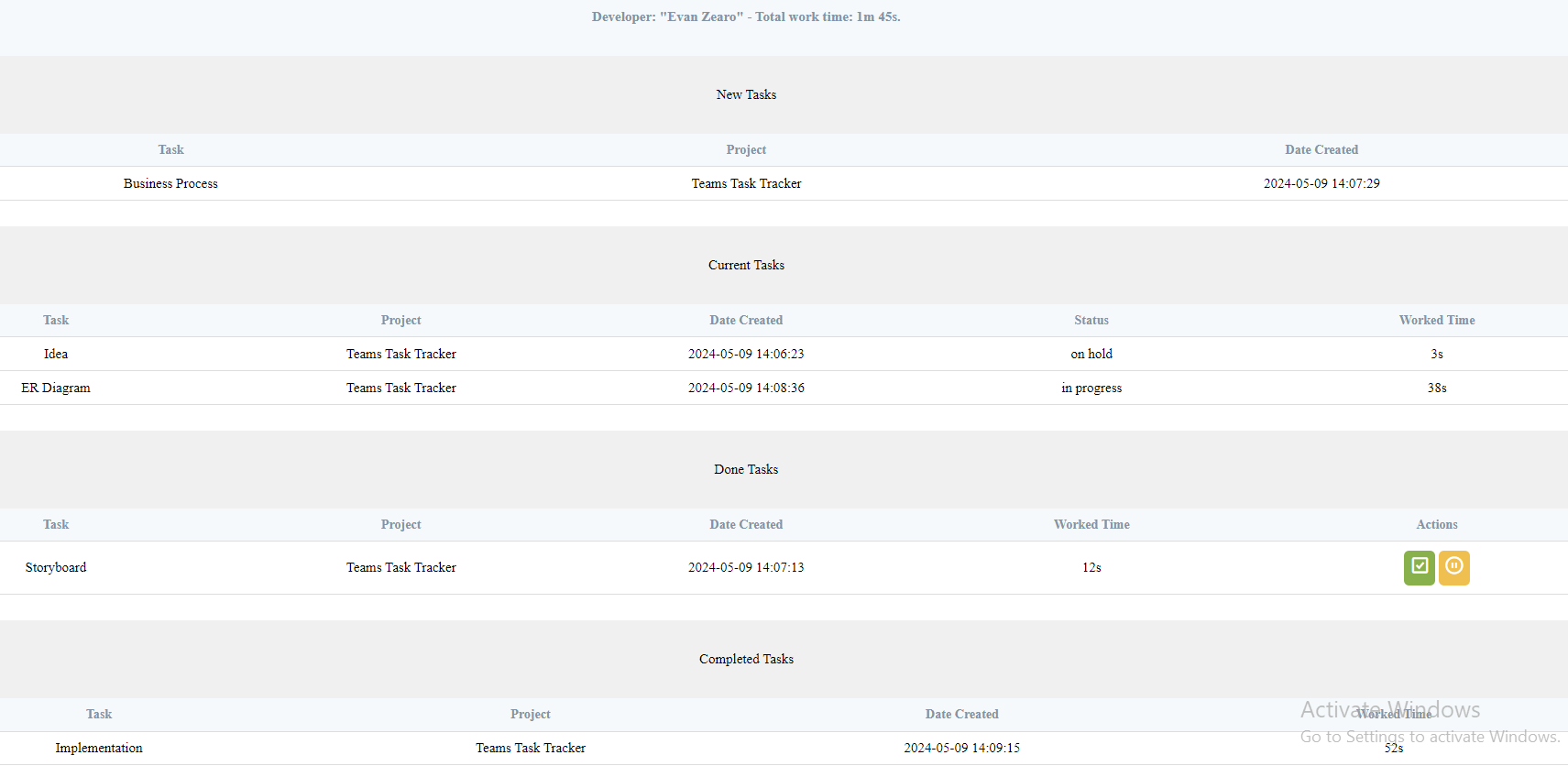
[Figure 12](#Figure12_leadersViewDevelopers) shows that in this page, the leader may see all of his developers’ general information as well as some information related to their tasks. The leader may as well search for developers by their names or emails for more precise seeking. Moreover, he/she may click on the show icon to see more details about a developer (See [figure 13](#Figure13_viewDevelopersTasks)) or the archive icon to archive a developer’s account. Furthermore, the leader may also click on the Add New button to create a new developer account. (See [figure 14](#Figure14_addDeveloper))



*Figure 12 – Leader’s View Developers*

##### 4.2.10- View Developer’s Tasks:

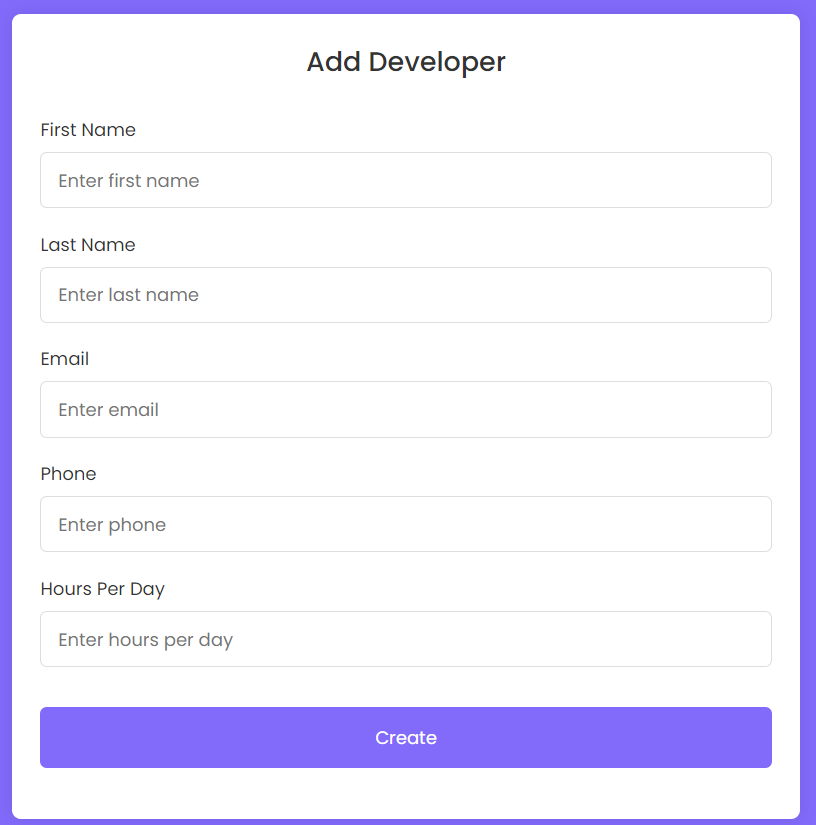
This page allows the leader to see all the tasks history of a chosen developer. History includes Newly assigned tasks, current tasks that are being worked on, tasks that the developer marked as done, as well as the tasks that the leader has marked as completed. The leader may also click on the check icon to mark a task as completed or the resume icon to return the task to the developer for additional work. (See [figure 13](#Figure13_viewDevelopersTasks))



*Figure 13 – View Developer’s Tasks*

##### 4.2.11- Add Developer Page

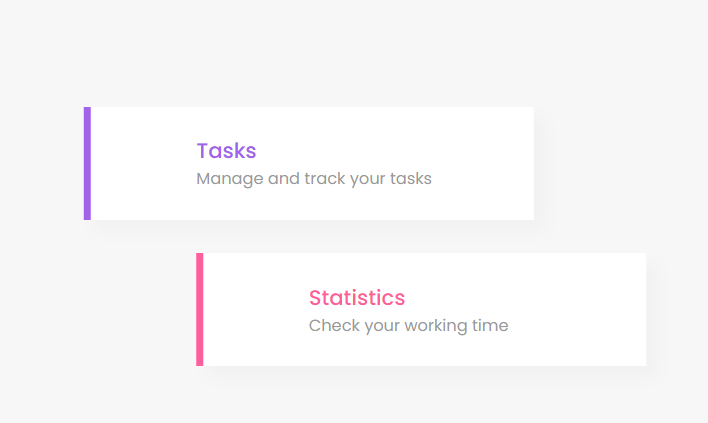
This page allows the leader to create a new developer account for the company. After creation, the system automatically sends an email to the given account to inform them of their account creation with their password. (See [figure 14](#Figure14_addDeveloper))



*Figure 14 – Add Developer*

##### 4.2.12- Developer’s Main page:

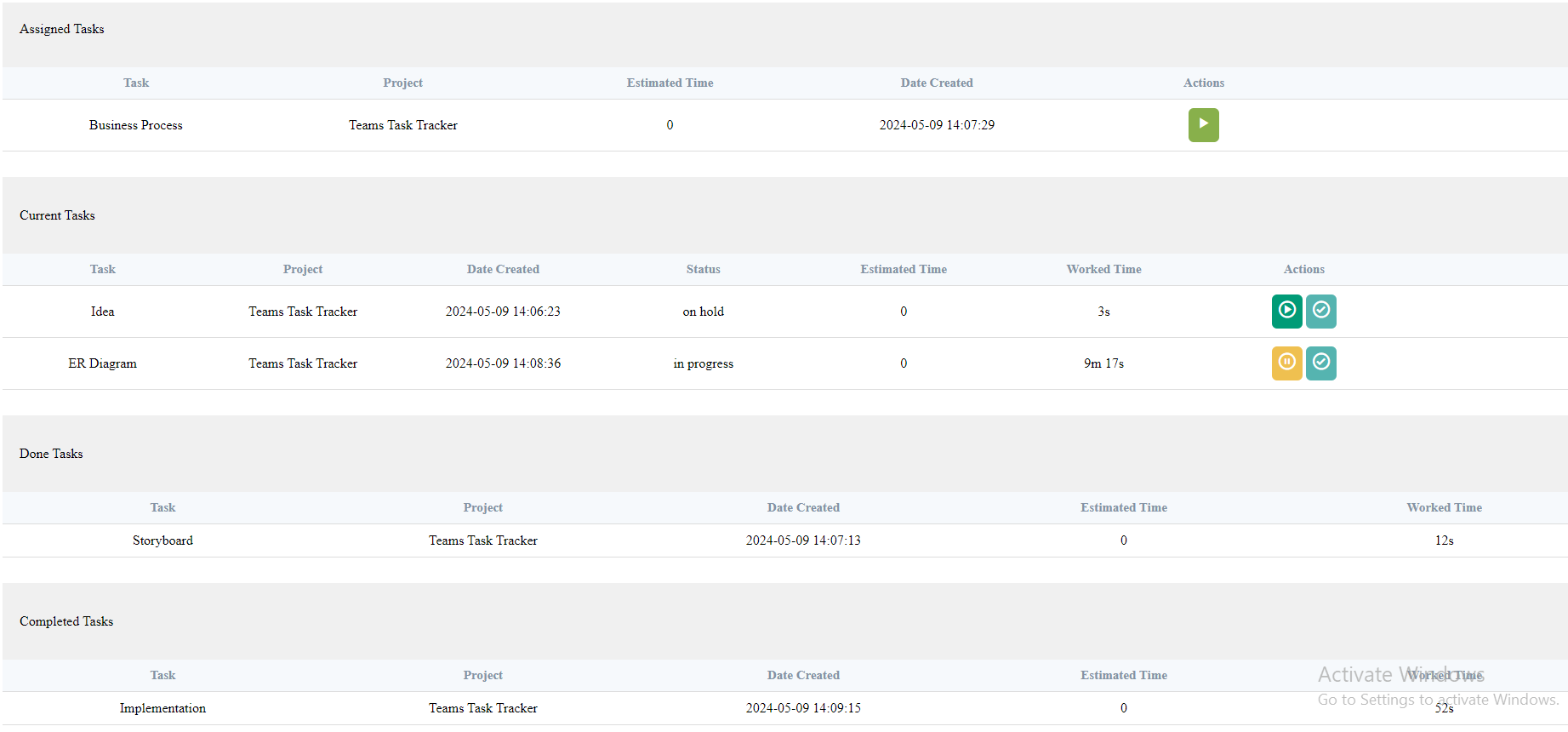
The first page that a developer sees when logging in is the developer’s main page. It contains a list of options to navigate through different pages. Every item in that list is a clickable link that navigates the developer to a different page accordingly. (See [figure 15](#Figure15_developersMain))



*Figure 15 – Developer’s Main Page*

##### 4.2.13- Developer’s Tasks Page:

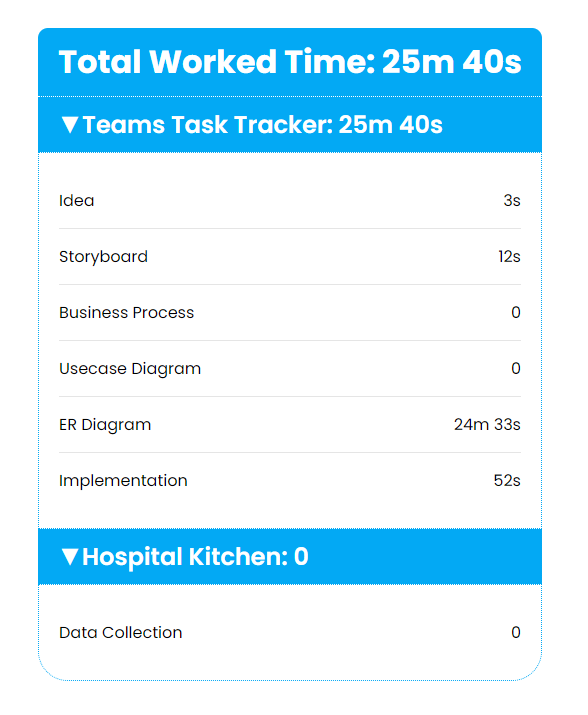
Here, the developer can check all of his tasks’ history along with some basic information. This includes newly assigned tasks, current tasks that are being worked on, tasks marked as “done” as well as tasks marked as “completed”. In addition, the developer may click the start icon to start a newly assigned task marking it as “in progress. Clicking the resume pause icon marks a task as “on hold” while the resume icon does the opposite by marking it as “in progress” again. Last icon is the Check icon which when clicked, marks the task as “done” for the leader to view. (See [figure 16](#Figure16_developersTasks))



*Figure 16 – Developer’s Tasks*

##### 4.2.14- Developer View Statistics Page:

In this page, the developer can check his/her own overall statistics in the company. The header shows the overall spent time in all tasks combined. Then, every project has its own block that shows how much time has every task taken with the project’s overall time combined. The developer may click the collapse (arrow) icon to toggle the view of the project for a desired view. (See [figure 17](#Figure17_developersStatistics))



*Figure 17 – Developer’s Statistics Page*

# Chapter 5

## Conclusion:

My aim was to plan and implement a flexible and easy to use online task manager and tracker for Computer Science based companies. Since after following some researches, I noticed that most developers now a day are working remotely. This, this website helps any similar company to work flexibly anywhere anytime.

## Future Work:

In the future. I am hoping to implement the ability for leaders to make a **Pert Chart** for a project. This will help in the automation aspect of the company for when the developers are working on the tasks since some tasks depends on others to be finished first. Thus, developers shouldn’t be able to work on any task before completing its prerequisite/s.

## References

*github*. Retrieved from github.com: https://github.com/

*stackoverflow*. Retrieved from stackoverflow.com: https://stackoverflow.com/

*youtube*. Retrieved from youtube.com: https://youtube.com/