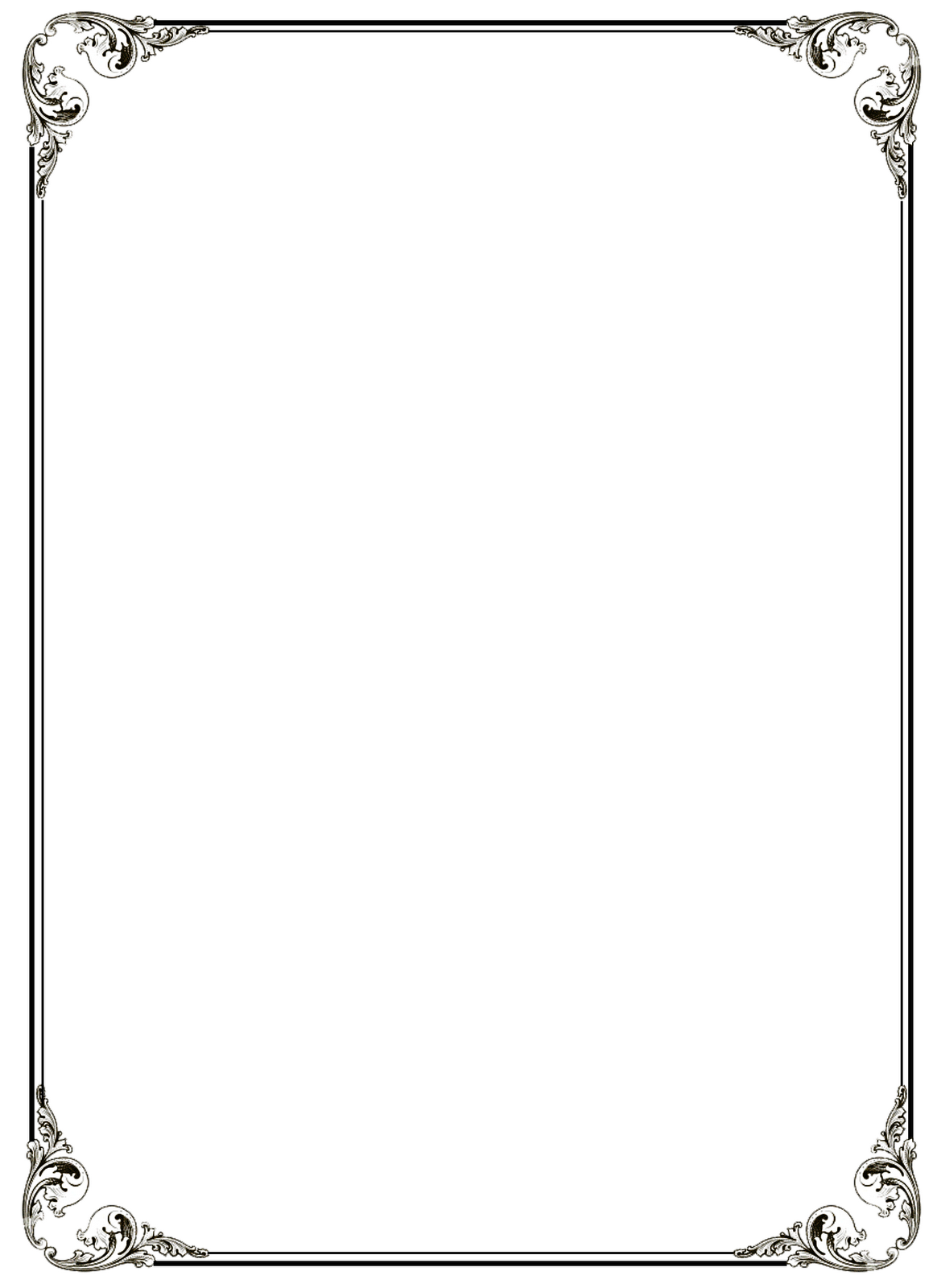


**Hyper Task**

|  |  |  |
| --- | --- | --- |
| **Supervisor:** | *NGO PHUOC NGUYEN* | |
| **Semester:** | *4* | |
| **Batch No:** | *T11908E0* | |
| **Group No:** | *02* | |
| **Order:** | **Full name** | **Roll No.** |
| *1* | *Duong Ngoc Hai* | *Student1198484* |
| *2* | *Nguyen Xuan Phong* | *Student1198583* |
| *3* | *Nguyen Hien Long* | *Student1198731* |
| *4* | *Le Tuan Khang* | *Student1204495* |

***Month****: 07* ***Year****: 2021*



This is to certify that

|  |  |
| --- | --- |
| **Mr.** | **Duong Ngoc Hai** |
| **Mr.** | **Nguyen Xuan Phong** |
| **Mr.** | **Nguyen Hien Long** |
| **Mr.** | **Le Tuan Khang** |

Have successfully Designed & Developed

|  |  |  |
| --- | --- | --- |
|  |  |  |
| **Hyper Task** | | |

Submitted by:

|  |
| --- |
| **Mr. Ngo Phuoc Nguyen** |

Date of Issue:

|  |  |
| --- | --- |
|  |  |

Authorized Signature:

|  |  |
| --- | --- |
|  |  |

# TABLE OF CONTENTS

[TABLE OF CONTENTS ii](#_Toc78832986)

[ACKNOWLEDGEMENTS iv](#_Toc78832987)

[PROJECT SYNOPSIS v](#_Toc78832988)

[REVIEW 1 1](#_Toc78832989)

[PROBLEM DEFINITION 2](#_Toc78832990)

[Project Requirement 2](#_Toc78832991)

[PROJECT DESIGNING 5](#_Toc78832992)

[1. Dashboard 5](#_Toc78832993)

[2. Boards 5](#_Toc78832994)

[3. Features 5](#_Toc78832995)

[4. User Page 5](#_Toc78832996)

[5. Register/Login 6](#_Toc78832997)

[6. About us 6](#_Toc78832998)

[HARDWARE / SOFTWARE REQUIREMENTS 6](#_Toc78832999)

[1. Server Requirement: 6](#_Toc78833000)

[2. Client Requirement: 6](#_Toc78833001)

[TASK SHEET REVIEW 1 8](#_Toc78833002)

[REVIEW 2 9](#_Toc78833003)

[I. Architecture & Design of the Project 10](#_Toc78833004)

[1. Presentation Tier: 10](#_Toc78833005)

[2. Business Logic Tier: 10](#_Toc78833006)

[3. Data Access Tier: 10](#_Toc78833007)

[II. Data flow diagram 11](#_Toc78833008)

[1. Define 11](#_Toc78833009)

[2. The main components of Context Diagram: 11](#_Toc78833010)

[3. Hyper Task context chart 13](#_Toc78833011)

[4. Data flow diagram and sequence diagram 14](#_Toc78833012)

[4.1. Login 14](#_Toc78833013)

[4.2. Register 15](#_Toc78833014)

[4.3. User 16](#_Toc78833015)

[4.4. Board 17](#_Toc78833016)

[4.5. List 19](#_Toc78833017)

[4.6. Card 21](#_Toc78833018)

[4.7. Activity 22](#_Toc78833019)

[III. Database design 23](#_Toc78833020)

[1. Introduction 23](#_Toc78833021)

[2. Entity Relation Diagram 24](#_Toc78833022)

[3. Main entity description 25](#_Toc78833023)

[4. Database entity detail 28](#_Toc78833024)

[4.1. User 28](#_Toc78833025)

[4.2. Board 28](#_Toc78833026)

[IV. Sitemap 31](#_Toc78833027)

[TASK SHEET REVIEW 2 32](#_Toc78833028)

[REVIEW 3 33](#_Toc78833029)

[1. Dashboard 34](#_Toc78833030)

[2. Contact Us 35](#_Toc78833031)

[3. Boards 36](#_Toc78833032)

[4. Features 38](#_Toc78833033)

[5. Login 39](#_Toc78833034)

[6. Register page 40](#_Toc78833035)

[7. Profile page 41](#_Toc78833036)

[8. User’s card page 42](#_Toc78833037)

[9. User’s activity page 43](#_Toc78833038)

[10. Contact Page 44](#_Toc78833039)

[TASK SHEET REVIEW 3 45](#_Toc78833040)

# ACKNOWLEDGEMENTS

The thirst for learning, upgrading technical skills and applying the concepts in real life environment at a fast pace is what the industry demands from IT professionals today. However busy work schedules, far-flung locations, unavailability of convenient time-slots pose as major barriers when it comes to applying the concepts into realism.

So this is what we need, the last challenge of our course is final eProject.

The Objective of this eProject is to give a sample project to work on real life projects. These applications help us build a larger more robust application.

Final we want to say special thanks for our families and friends that always support and encourage us during the time we work with this project.

Last words to our mentor: Nguyen Ngo Phuoc. Thanks for always being a good mentor and for guiding us on the right path. We will always be thankful to you.

# PROJECT SYNOPSIS

Employees needed a tool to easily connect, collaborate, and do more together in an increasingly distributed company.

Many people may worked in different countries and timezones, complicating collaboration—especially hard if there wasn’t a single source of information for everyone to go to.

Coordinating work across teams is chaotic—especially if you have to dig through emails or messaging apps to find information. Without a system for planning, organizing, and executing work, confusion and chaos can grow. That’s where work management can help.

Our’s goal was to ensure a smooth, organizanization-wide digital transformation for how employees all work together. A critical piece of this digital transformation was implementing a work management platform.

Work management is a systematic approach to orchestrating an organization’s workflows—be it a project, an ongoing process, or routine tasks—to provide the clarity teams need in order to hit their goals faster. It’s about coordinating people and work across all levels of an organization to ensure that everyone has the information they need to accomplish the work that matters most.

Those services will be offered by Hyper Task, work management app that brings clarity to teams, giving them the confidence to blaze ahead with the most important work and stop second guessing or running into unseen roadblocks.

**REVIEW 1**

# PROBLEM DEFINITION

Project Requirement

Our website - Hyper Task should work towards enhancing user’s efficiency and improve user’s productivity, not add on to the existing workload…

To achieve the above goals, our website need some features:

* Task Planning

Effective task management starts with appropriate planning. Depending on the budget, resources, scope, and timeline; efficient task planning requires breaking down the work into stages through a structured approach.

With a good task management tool, you should be able to break down your work into simpler tasks that can be easily managed. Moreover, there should be an option to organize and prioritize tasks as required.

* Task Scheduling

The idea behind task management is to organize work, be it personal or professional, on a daily basis to make it more streamlined.

A good management tool should allow for creating and managing schedules. When work is planned on a daily, weekly or monthly basis, it becomes easier to not only miss out on important tasks but also helps monitor progress. Notifications and reminders make task scheduling even better as they keep you on alert and can save you from overspending much time on a single task.

* Recurring Tasks

Another option most good management tools have is the option to create a task and schedule it on multiple days for various purposes.

Some tasks need to be repeated daily, e.g. your daily alarm and others occasionally, e.g. your doctor's appointment.

When it comes to professional projects, a single task can at times serve multiple projects. For instance, the task of updating timesheets by the end of the week can be scheduled on a daily, weekly, or monthly basis for multiple teams.

* Priority

With so many things on the list, it is only but natural to get confused in sorting out tasks on priority. An efficient task management software can help with that by allowing the option for prioritizing tasks. Tasks are usually prioritized based on the urgency level of the task itself or the dependency of other tasks.

A good tool should be able to provide a mechanism to specify such priorities along with the tasks. This helps in completing urgent work first and speeding up implementation of the critical stages of the project.

* Task Tracking

When a number of tasks are scheduled, especially in teams, keeping track of all the tasks done can prove a struggle. In order to make this process easier, it is wise to adopt a tool that can help you trace exactly which activities are started, in progress and done with.

With properly defined tasks in terms of team members they are assigned to, the project they fall under, and their urgency level, you can get the added benefit of searching and filtering out tasks using any of the criteria. When you can track tasks, you can make task and project planning easier while saving time on the unnecessary recreation of tasks.

* Calendar

A good task management tool should be easily integrated with a simplified calendar that is easy to manage and access. Through a calendar, you can organizing and schedule tasks as well as use it to work for project timelines and meetings.

Whether you need to mark deadlines on the calendar to help team easily follow up on the work progress or you need to schedule meetings and conferences, a central calendar is vital.

* Collaboration

In teams, collaboration goes hand in hand with task management. Which is why we believe that a task management tool should be designed to work toward improving team collaboration. Complicated or underperforming apps can greatly impact communication, which can in turn negatively affect task management.

An efficient collaboration channel is a key element to look for in a task management platform. The team should be able to exchange information, share tasks and receive task updates. Be it file attachments or elaborating on assigned tasks, the team should be able to communicate information on time and with ease.

* Time Tracking

A core element of streamlining work is to keep track of the time for each activity. This way you can measure the time spent to do each task, whether it's worth the time and how contributes to or has an overall impact on the project, in terms of budget and time.

When working with teams, the importance of tracking time manifests itself in 2 ways: to calculate the time spent by the resource on each task and to calculate payroll based on the hours worked. The hours are then logged into timesheets that need to be reviewed by the management. Reporting and Visualization

A part of task management is to monitor and observe task progress. This is needed to regulate the flow of tasks so as to avoid missing deadlines and making task progress more effectively.

A good software should be able to provide functionalities that help easily monitor as well as report progress for tasks.

**PROJECT DESIGNING**

1. Dashboard

Introduction of the website with hyperlink at the top (for navigation). Dashboard gives a current summary, in graphic, easy-to-read form, of key information relating to progress and performance, especially of a board or a project.

1. Boards

Containts below features:

* Search Board: user can search for their board
* Follow Board: user can watch/delete all their followed boards
* Boards List: user can watch/delete all board they created and assigned
* Add Board: user can create their new board
* When click board, user will be lead to a single board working area:
  + Board: User can edit name, background, add to followed boards
  + List: User can create/edit/delete working list and create card
  + Card: User can edit/delete working card
  + Add Member: User can add more member to this board

1. Features

Features page includes various introduction of website to give users an overview of the features and what can be done with Hyper Task

1. User Page

Contains customer information:

* Personal information: user can view their profile and change some field.
* Activity: activity of user.
* Contact: display a list of user’s contact, user can also find and add contact here.
* Cards: display a list of user’s card, these cards may be created by them or assigned to them, so they can track the status of them.

1. Register/Login

* Register: Allows users to create accounts to perform system functions when users do not have an account.
* Log in: After logging in, the user can use the website's functions.

1. About us

Brief information of the store and some communication method.

# HARDWARE / SOFTWARE REQUIREMENTS

1. Server Requirement:

* **Hardware**
* AMD Ryzen/ Intel core i3 3.0 Ghz or better
* RAM: 4 GB or better
* HDD: free space at least 2 GB or better
* **Software**
* Java IDE (eclipse EE/IntelliJ)
* Visual studio code
* RBDMS: MySQL workbench
* NodeJS version 12.x or higher
* NPM version 5.x or higher
* Web Browser: Chrome, Opera, Firefox (lastest will be the best)
* **Technology**
* ReactJS, Redux, Spring Boot, MySQL, Hibernate, v.v…

1. Client Requirement:

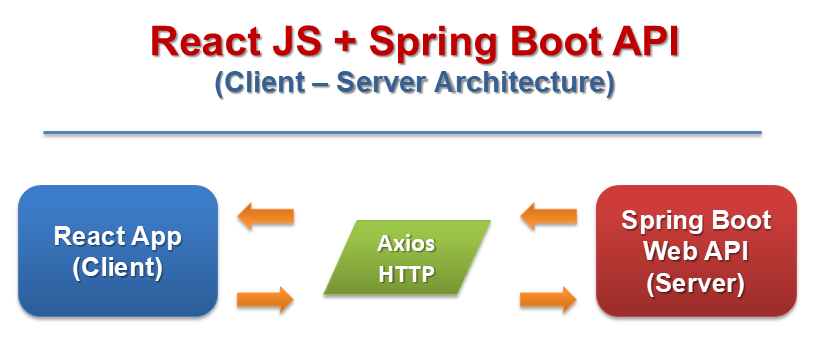
* **Hardware**
* AMD Ryzen/ Intel core i3 3.0 Ghz or better
* RAM: 4 GB or better
* HDD: free space at least 1 GB or better
* **Software**
* Web Browser: Chrome, Opera, Firefox (lastest will be the best)

**TASK SHEET REVIEW 1**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Members** | | **Date Preparation Of Activity Plan** | | | |
| **No** | **Prepared by** | **Task Description** | **Start Date** | **End Date** | **Status** |
| 1 | All Members | Acknowledgment/Synopsis | 15/07/2021 | 18/07/2021 | Completed |
| 2 | Problem Definition | Completed |
| 3 | Project Designing | Completed |
| 4 | Hardware/Software | Completed |
|  | | | | | |
| **Teacher** | | | **Team Leader** | | |
|  | | |  | | |
| **Nguyen Ngo Phuoc** | | | **Duong Ngoc Hai** | | |

**REVIEW 2**

1. Architecture & Design of the Project



* 1. Presentation Tier:

Is the tier in which the users interact with application. In React presentation Tier contents Presentational and container components.

Technology: ReactJS, Redux, Axios,…

* 1. Business Logic Tier:

Is mainly working as the bridge between Data Tier and Presentation Tier. All the Data passes through the Business Tier before passing to the Presentation Tier.

Technology: Spring Boot Web API.

* 1. Data Access Tier:

Is basically the server which stores all the application’s data .Data tier contents Database Tables, Database Views and other means of storing Application Data .

Technology: My SQL, Hibernate.

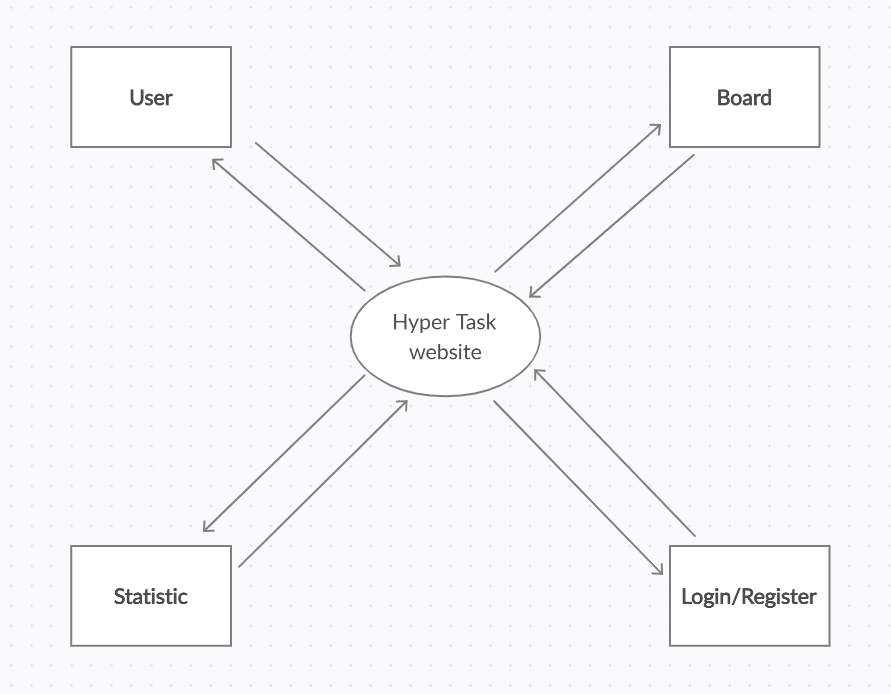
1. Data flow diagram
   1. Define

Data Flows Diagram (DFD) describes the information flow in the system. The next step of system analysis is to consider in detail the information necessary for the implementation for functions discussed above and the one necessary for the improvement of the functions. Modelling tool frequently used for this purpose is DFD. DFD will support 4 main activities:

* ***Analysis***: DFD is used to determine requirement of users.
* ***Design***: DFD is used to map out plan and illustrate solution to analysis and users while designing a new system.
* ***Communication***: One of the strength of DFD is its simplicity and ease to understand to analysts and users.
* ***Document***: DFD is used to provide special description of requirement and system design. DFD provide an overview of key functional components of the system but it does not provide any detail on these components. We have to use other tools like database dictionary, process specification to get an idea of which information will be exchanged and how.
  1. The main components of Context Diagram:

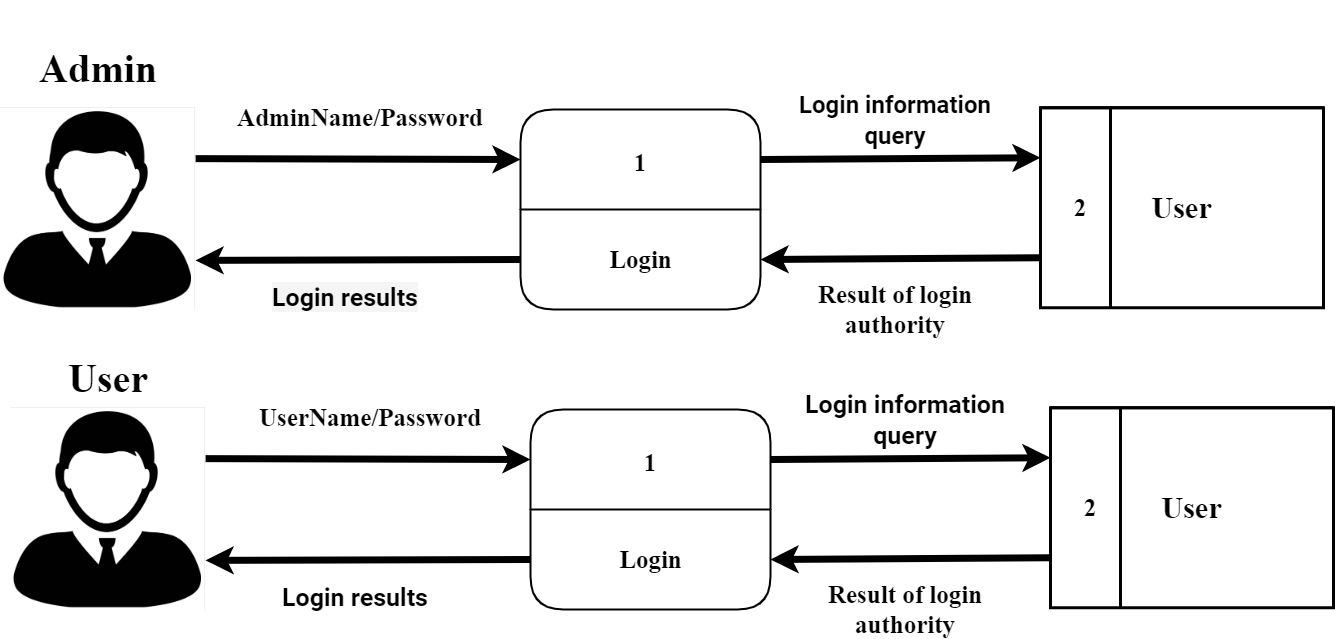
|  |  |  |
| --- | --- | --- |
| **Components** | **Illustration** | **Mô tả chi tiết** |
| **The Subject** | **User**  Không có mô tả. | The subject: External factors can be a person, a group of persons or an organization that are sources of information for the systems and are where system products are transferred to. |
| **The Process** | 1  Process | The common function of system |
| **The Data Flow** |  | Describes the movement of information from one part of the system to another. |
| **The Data Store** | 1 Data store | The data store: The Data Store is used to model a collection of data. A store is represented graphically by two parallel lines. The name of a Data Store that identifies the store is the plural of the name of the packets that are carried by flows into and out of the Data Store |

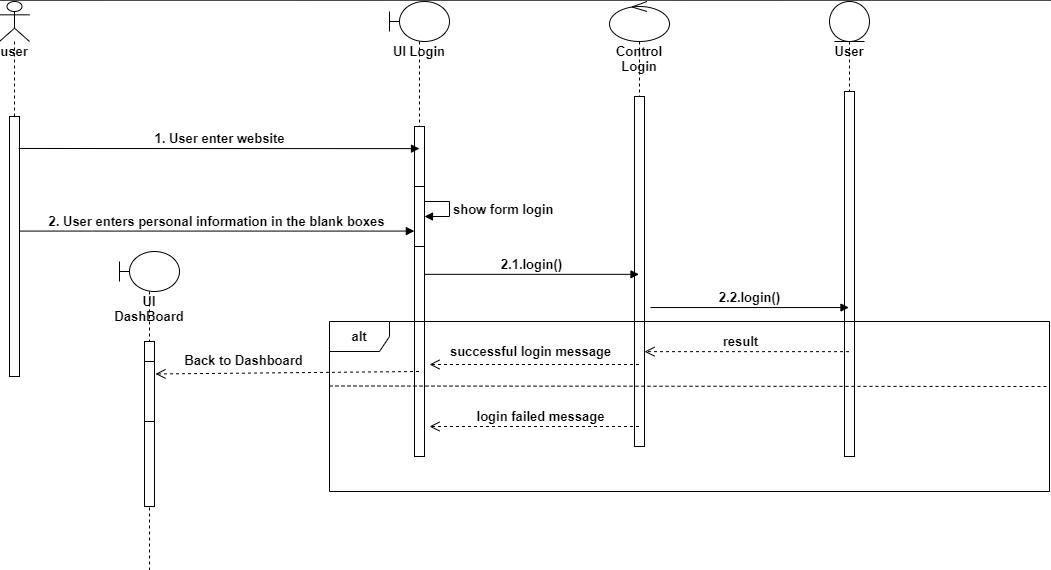
* 1. Hyper Task context chart



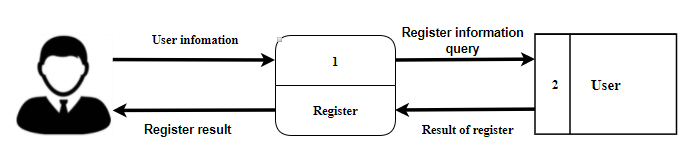
* 1. Data flow diagram and sequence diagram

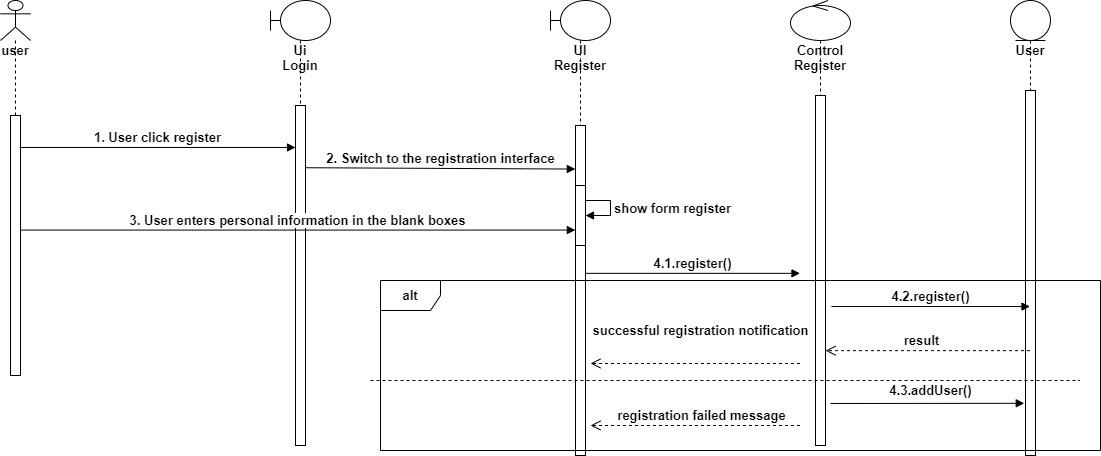
1. Login



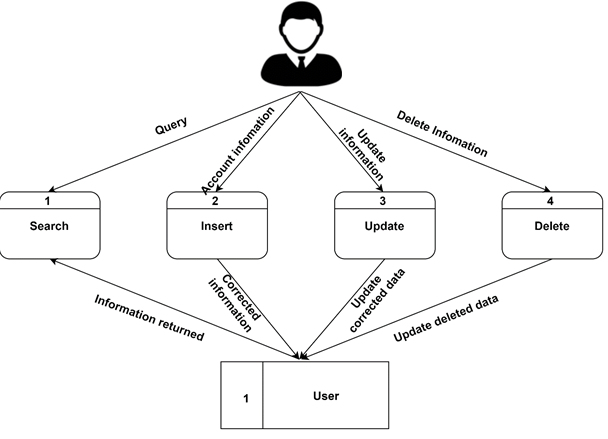


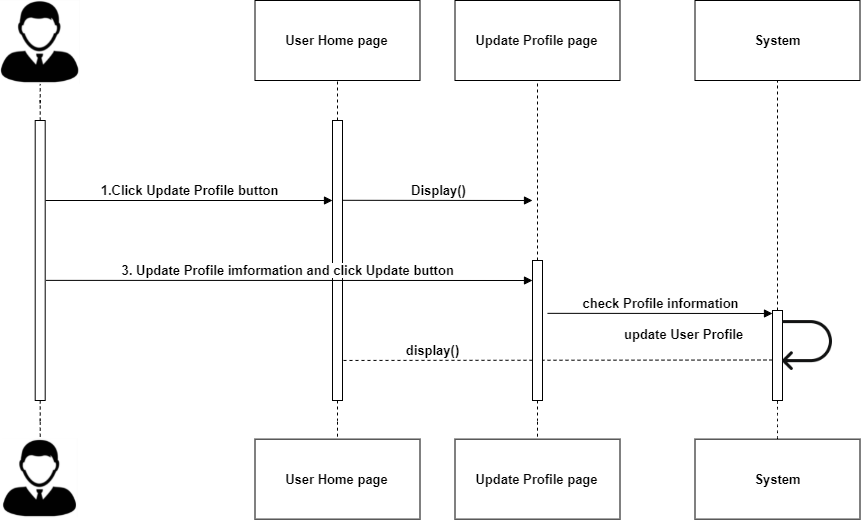
1. Register



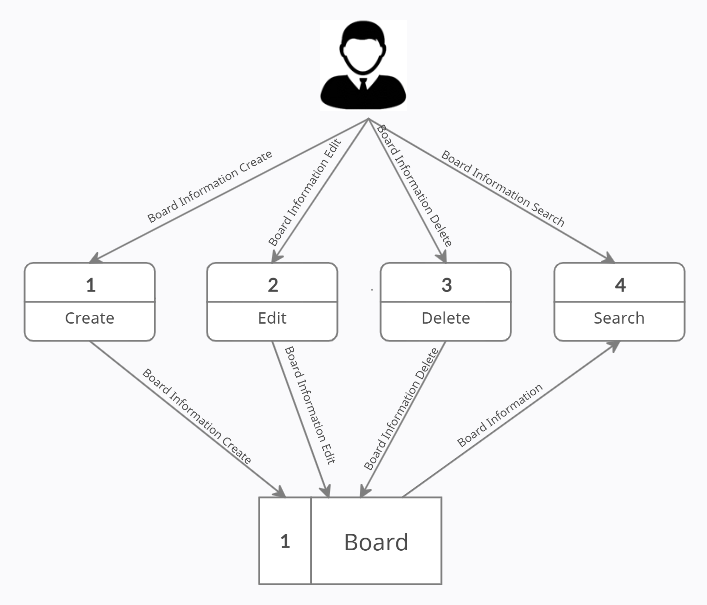


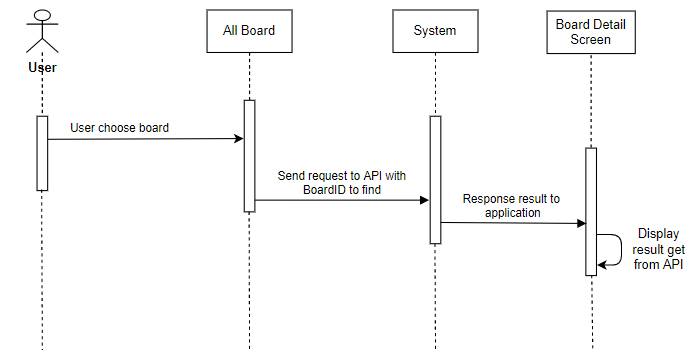
1. User

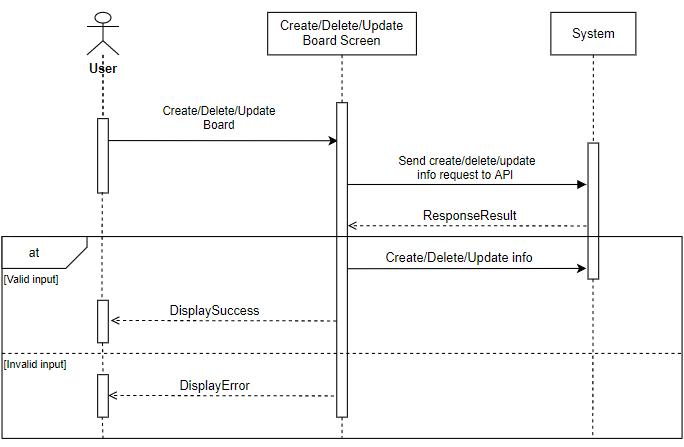




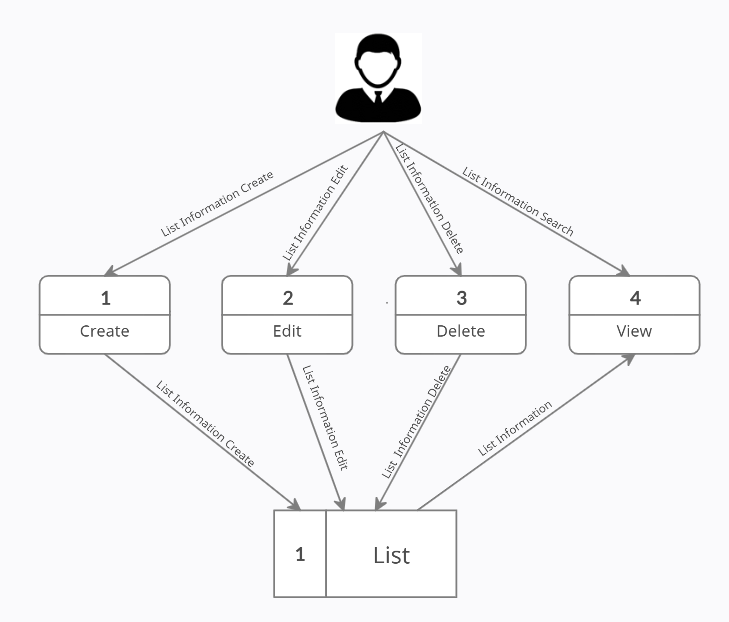
1. Board

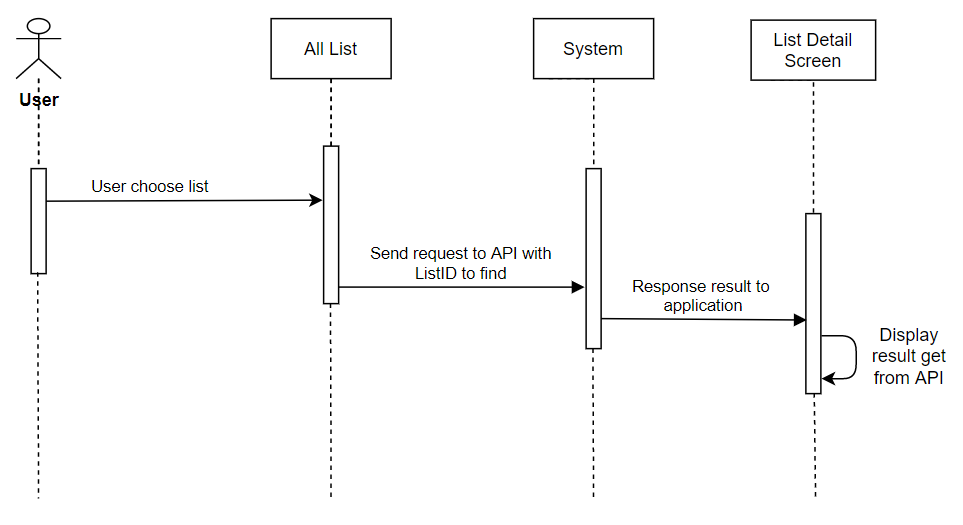


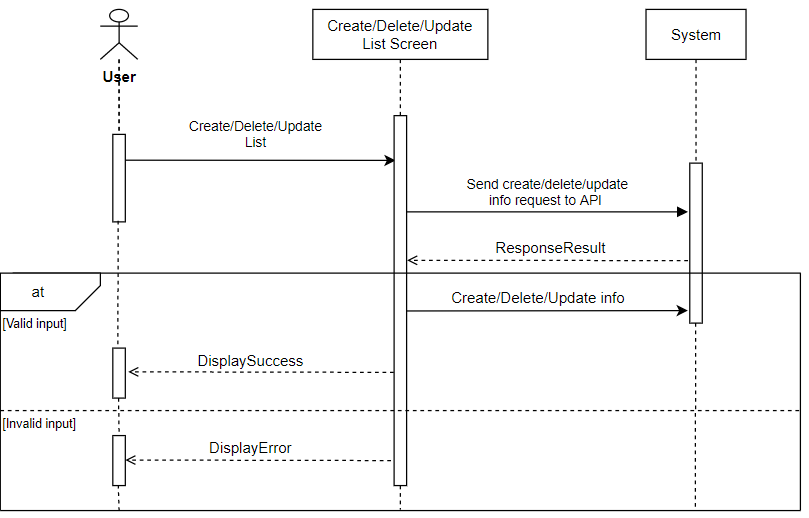




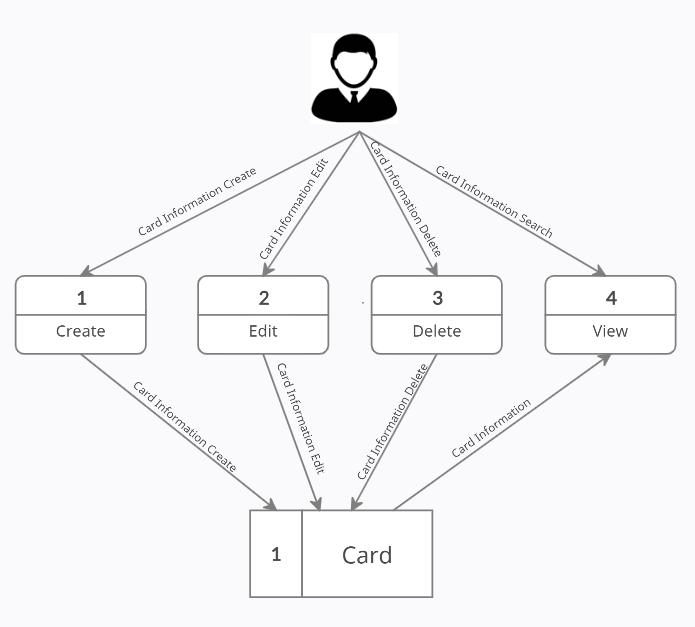
1. List

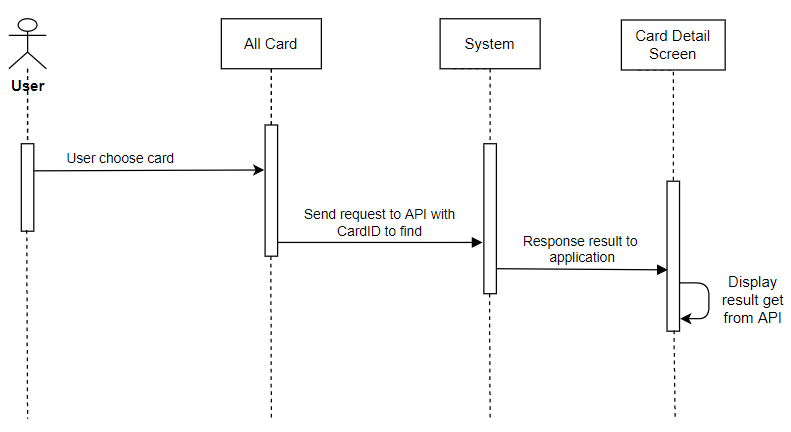




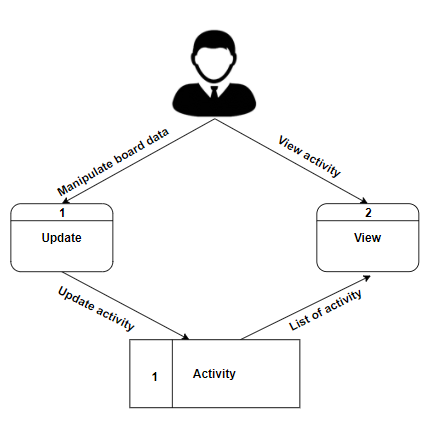


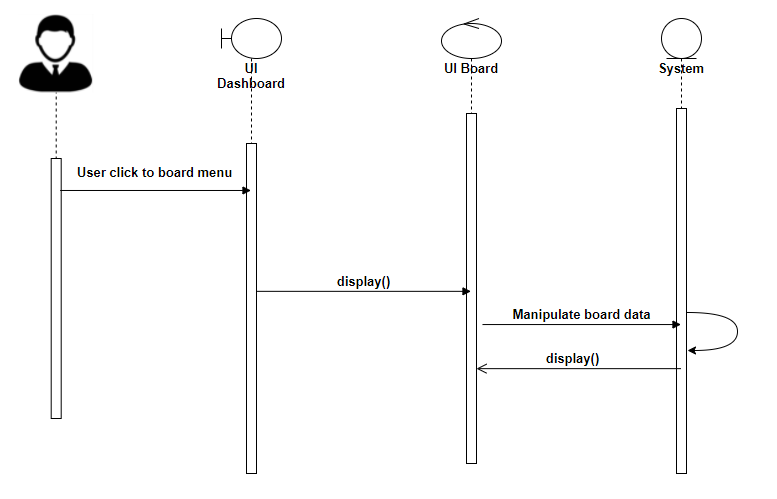
1. Card





1. Activity





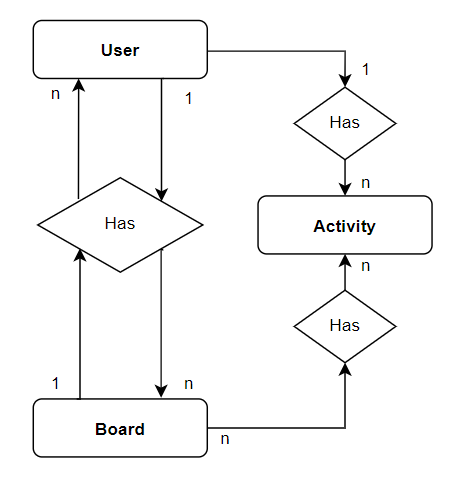
1. Database design
2. Introduction

The database is built in mySQL but its has archetype like NoSQL, most of object and relational will be nested in a stringified JSON.

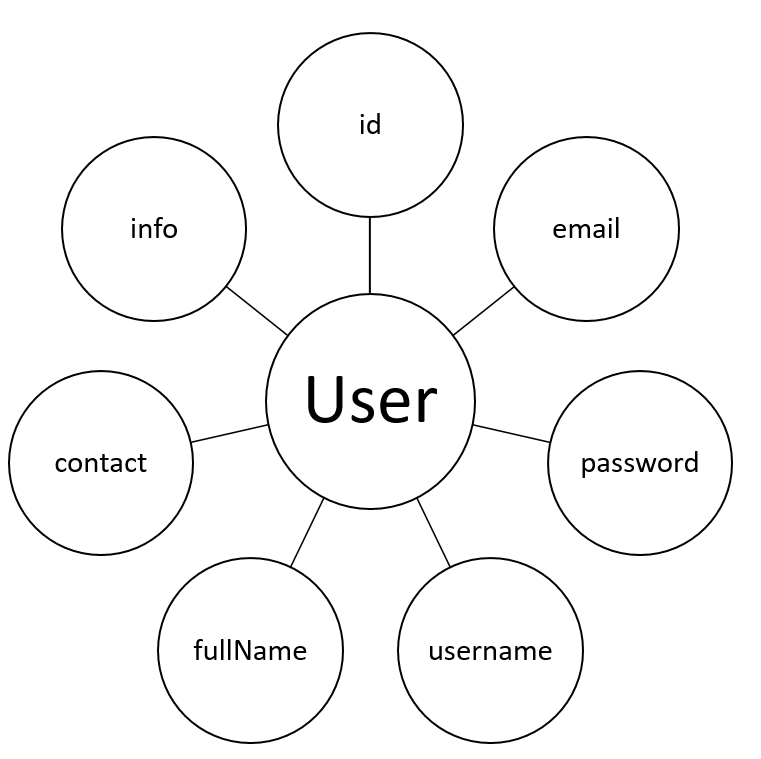
We choose is archetype to take some advantage of NoSQL

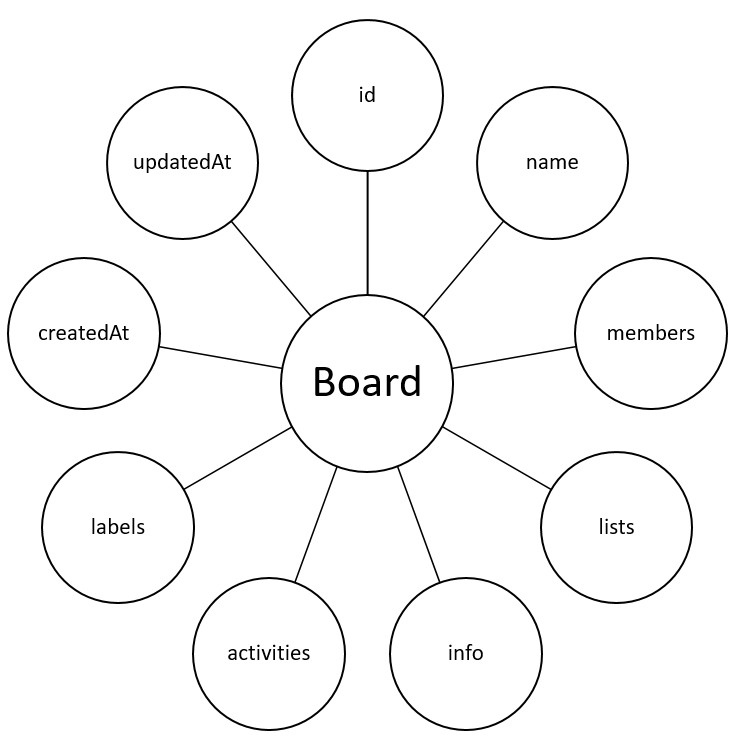
* **Flexibility**: this approach typically offer flexible diagrams that make development faster and more repeatable. The flexible data model makes NoSQL databases the ideal choice for unstructured or incompletely structured data.
* **Scalability**: databases are typically designed to scale using clusters of distributed hardware rather than scale up with expensive and powerful server additions. Some cloud service providers treat these activities non-publicly as a fully managed service.
* **High performance**: this approach are optimized for specific data models and access patterns which provide greater performance gains than trying to achieve the same level of functionality with a database relationship.
* **Extremely practical**: this databases provide extremely practical APIs and data types built specifically for each respective data model.

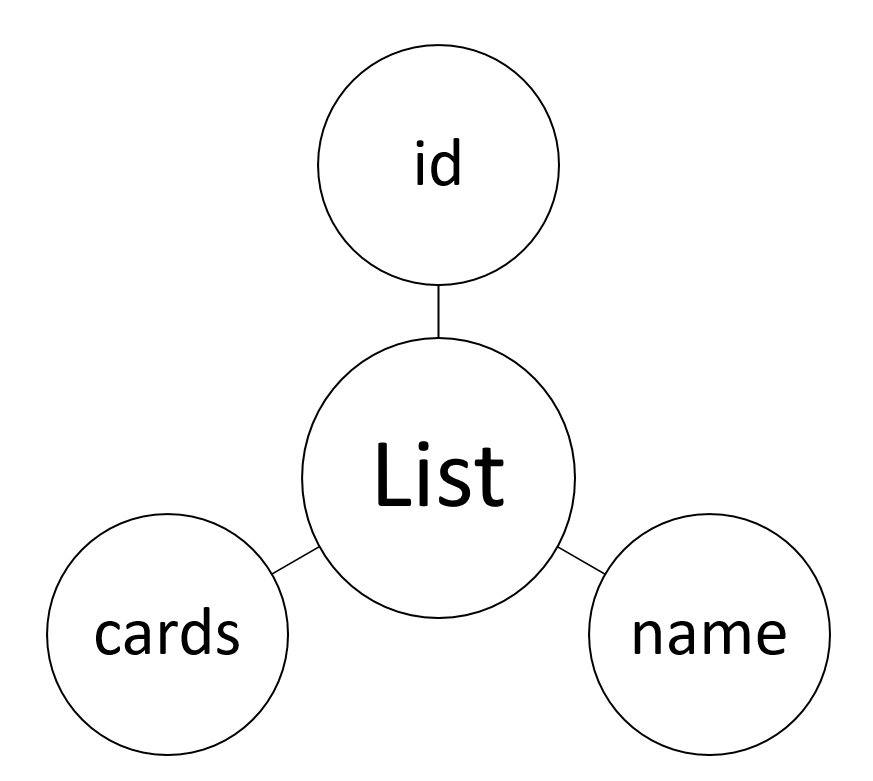
1. Entity Relation Diagram

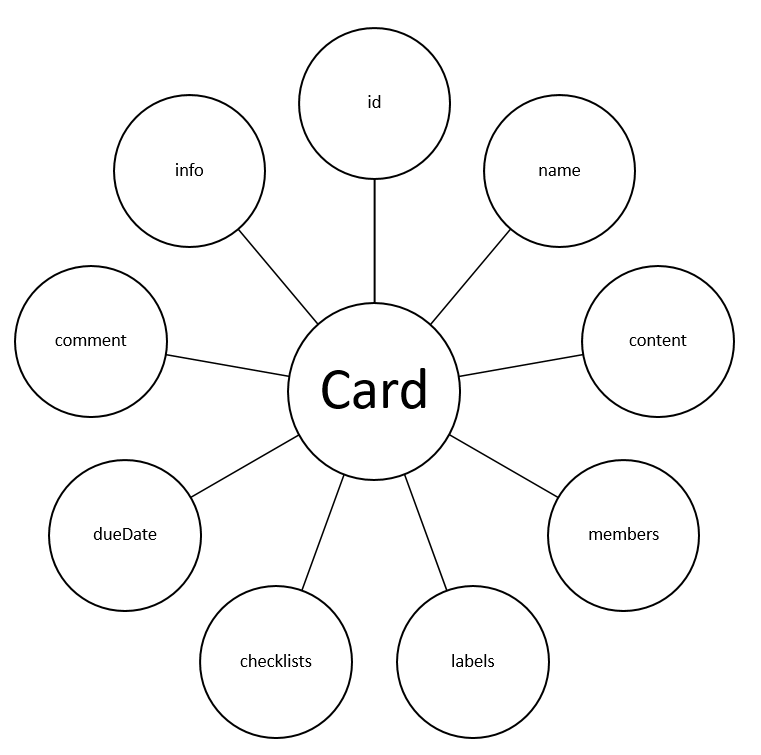
 

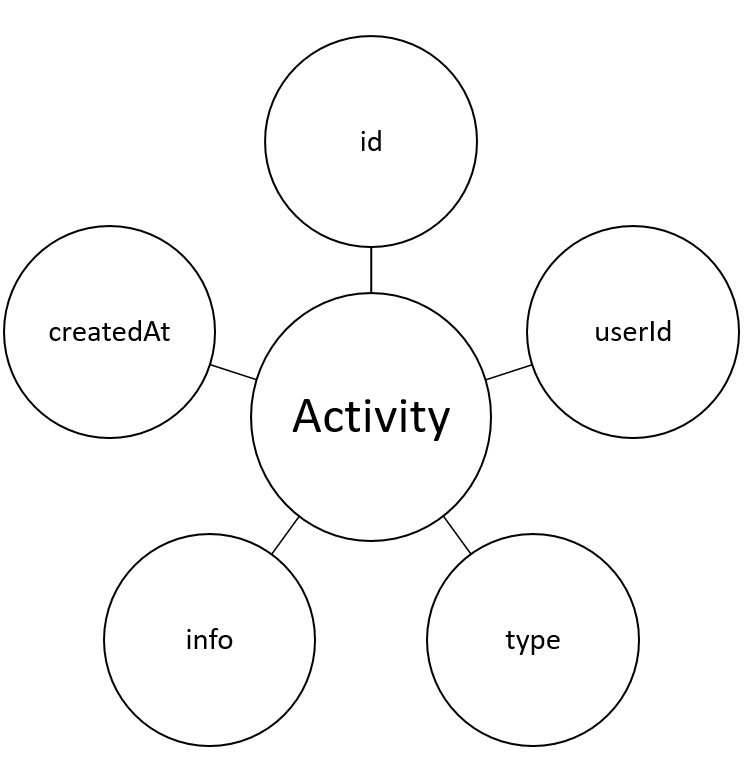
1. Main entity description











1. Database entity detail
2. User

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **User** | | | | | | |
| Name | Data Type | Primary  Key | Identity | Not  null | Default  value | Description |
| **id** | **int** | X | X | X |  | Id of User |
| email | varchar(50) |  |  | X |  | Email of User |
| password | varchar(50) |  |  | X |  | Password of User |
| username | varchar(50) |  |  | X |  | Username of User |
| fullName | varchar(50) |  |  | X |  | Name of User |
| contact | text |  |  |  |  | Stringified JSON contains multiple user |
| info | text |  |  |  |  | Stringified JSON contains multiple data |

User’s detailed property in json format:

"user": {

        "id": "",

        "username": "",

        "password": "",

        "email": "",

        "fullName": "",

        "contact": ["userId1", "userId2"],

        "info": {

            "avatar": "",

            "phoneNumber": "",

…

        }

    }

1. Board

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Board** | | | | | | |
| Name | Data Type | Primary  Key | Identity | Not  null | Default  value | Description |
| **id** | **int** | X | X | X |  | Id of User |
| name | varchar(50) |  |  | X |  | Email of User |
| members | varchar(50) |  |  | X |  | Password of User |
| lists | varchar(50) |  |  | X |  | Username of User |
| info | text |  |  | X |  | Stringified JSON contains multiple data |
| activities | text |  |  | X |  | Stringified JSON contains multiple data |
| labels | text |  |  | X |  | Stringified JSON contains multiple data |
| createdAt | datetime |  |  | X |  | Date created |
| updatedAt | datetime |  |  | X |  | Date updated |

Board’s detailed property in json format:

"board":{

        "id": "",

        "name": "",

        "members": [

            {

                "userId": "",

                "role": ""

            }

        ],

        "lists":[

            {

                "id": "",

                "name": "",

                "cards": [

                    {

                        "id": "",

                        "name": "",

                        "content": "",

                        "members":[],

                        "labels": [],

                        "checklist":[

                                {

                                "id": "",

                                "name": "",

                                "done": ""

                            }

                        ],

                        "dueDate": "",

                        "comment":[

                            {

                                "userId": "",

                                "content": "",

                                "date": ""

                            }

                        ],

                        "coverImage":""

                    }

                ]

            }

        ],

        "info": {

            "backgroundImage": "",

            "author": "authorId",

            "type": "personalBoard/template"

        },

        "activities": [

            {

                "userId": "",

                "type": [

                    "joined/removed from board",

                    "create/detele list",

                    "create/delete card",

                    "comment",

                    "add flags",

                    "set dueDate",

                    "add labels",

                    "..."

                ],

                "cardId": "",

                "listId": "",

                "date": ""

            }

        ],

        "labels": [

            {

              "id": "",

              "name": "",

              "color": ""

            }

        ],

        "author": "authorId",

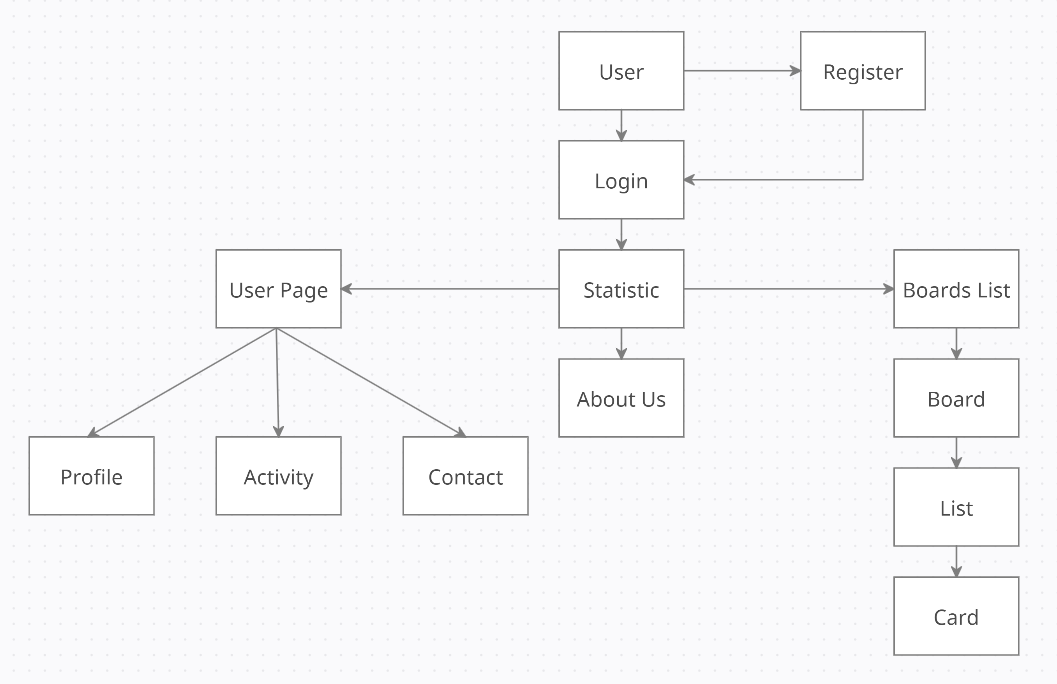
        "type": "personalBoard/template",

        "createdAt": "",

        "updatedAt": ""

    }

1. Sitemap

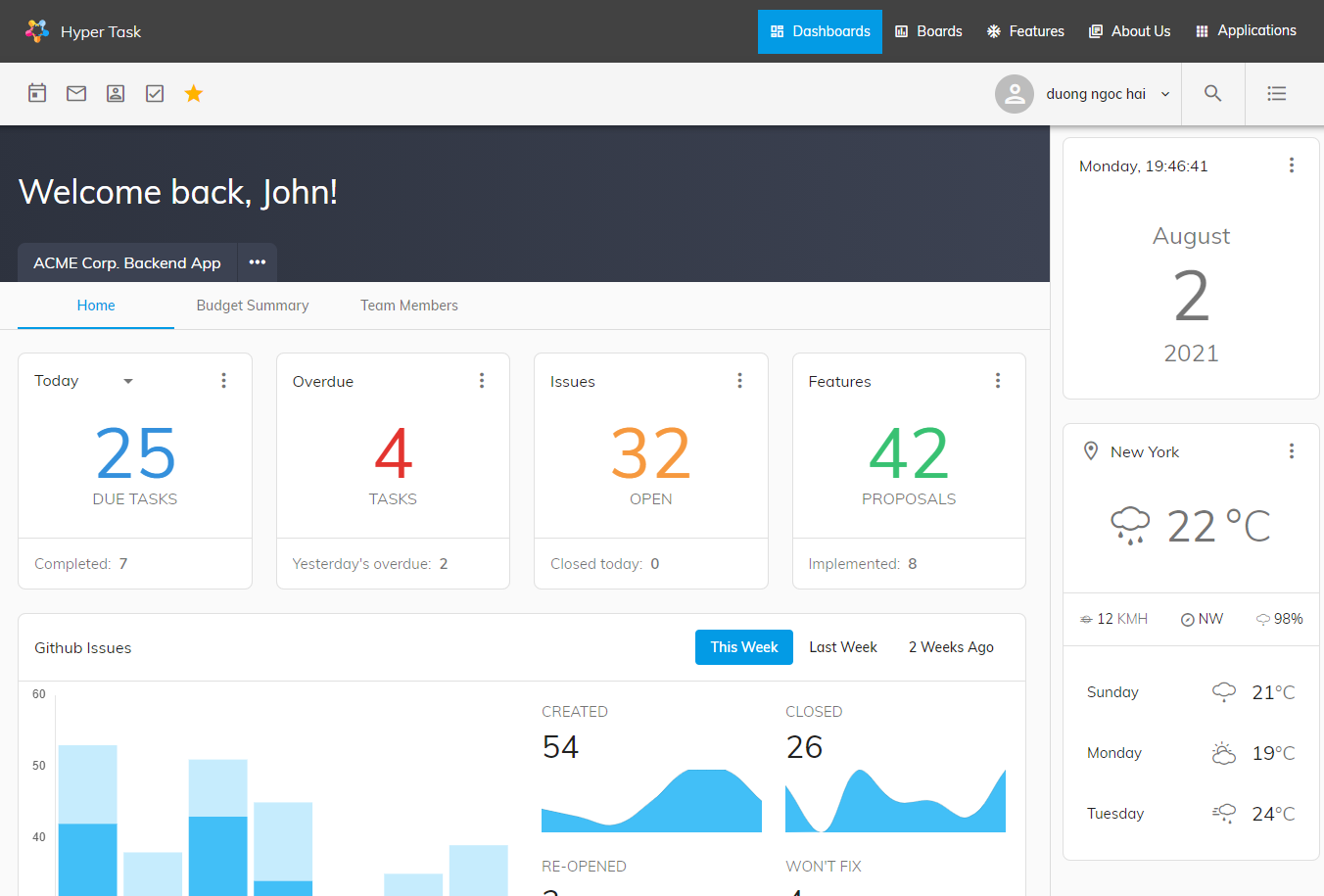


# TASK SHEET REVIEW 2

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Group Members** | | **Date Preparation Of Activity Plan** | | | |
| **No** | **Prepared by** | **Task Description** | **Start Date** | **End Date** | **Status** |
| 1 | Dương Ngọc Hải | Architecture & design of the project, database design, sequence diagram, coding | 16/07/2021 | 24/07/2021 | Completed |
| 2 | Nguyễn Hiền Long | Data flow diagram, coding, sequence diagram, Sitemap | Completed |
| 3 | Lê Tuấn Khang | Data flow diagram, coding sequence diagram | Completed |
| 4 | Nguyễn Xuân Phong | Data flow diagram, coding, sequence diagram | Completed |
|  | | | | | |
| **Teacher** | | | **Team Leader** | | |
|  | | |  | | |
| **Nguyen Ngo Phuoc** | | | **Duong Ngoc Hai** | | |

**REVIEW 3**

1. Dashboard

****

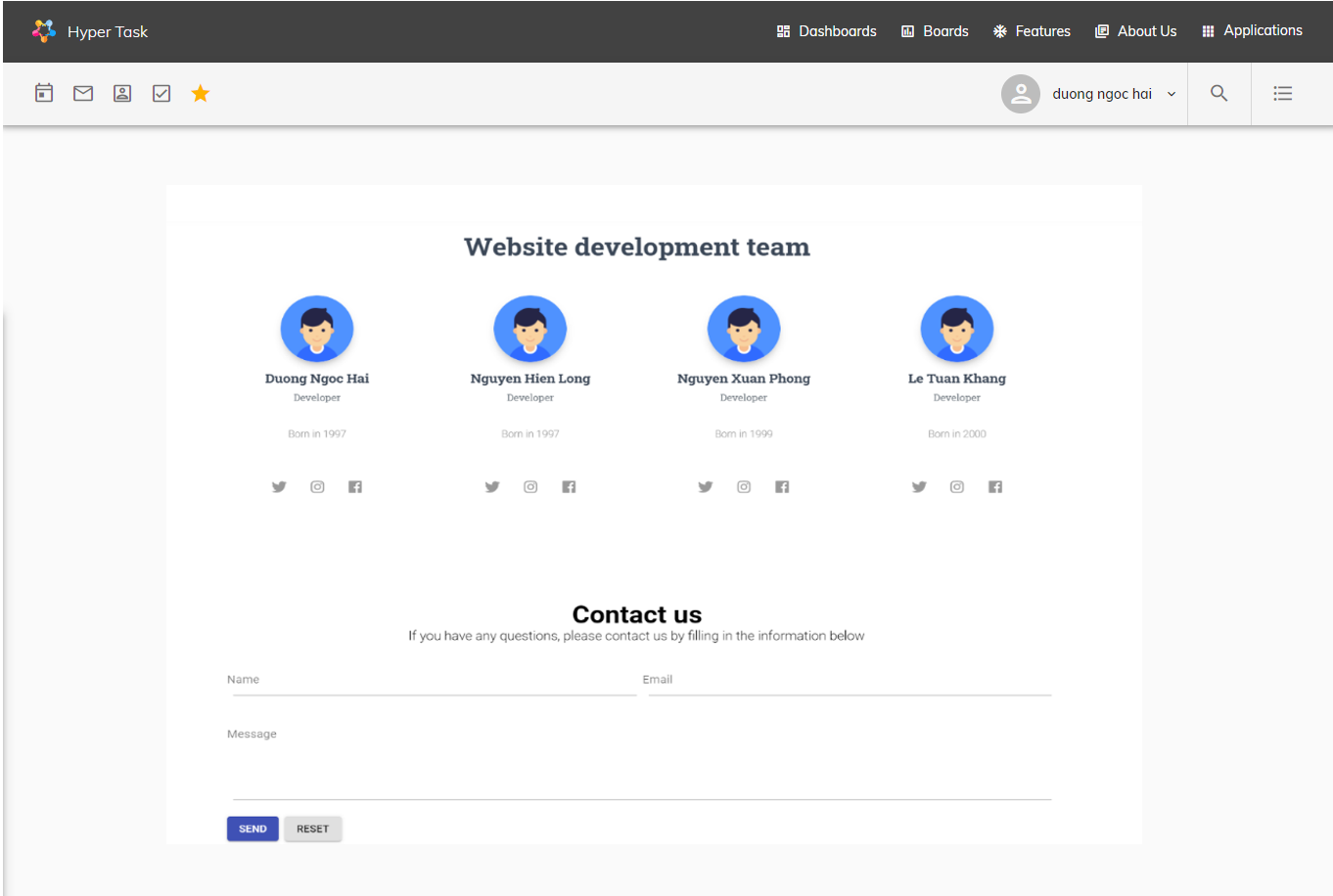
**Purpose:** Home page, to see statistic of user’s projects and other information.

**From the page**: when accessing the link http://www.localhost:3000/dashboard

To page: All links in header-menu and footer-menu link to other pages.

**Description**: This is a dynamic page, retrieving data from board and user tables. Introduction of the website with hyperlink at the top (for navigation). Dashboard gives a current summary, in graphic, easy-to-read form, of key information relating to progress and performance, especially of a board or a project.

1. Contact Us



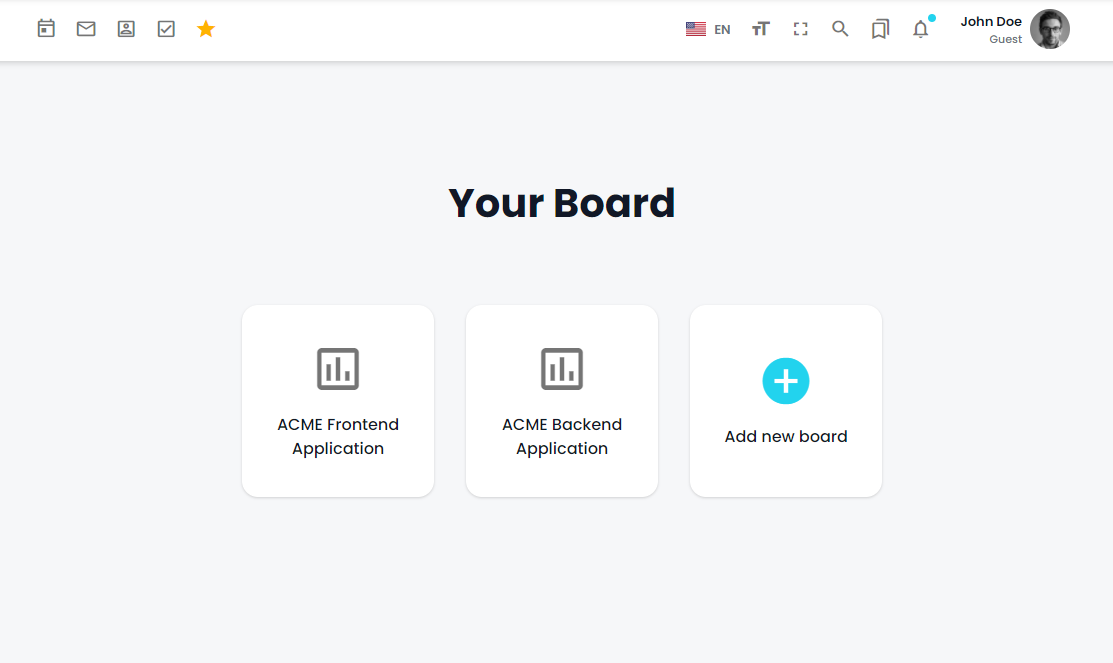
**Purpose**: Introducing the Hyper Task team members

**From page**: Home – About Us

**To page**: All links in header-menu and footer-menu link to other pages.

**Description**: This is a dynamic page

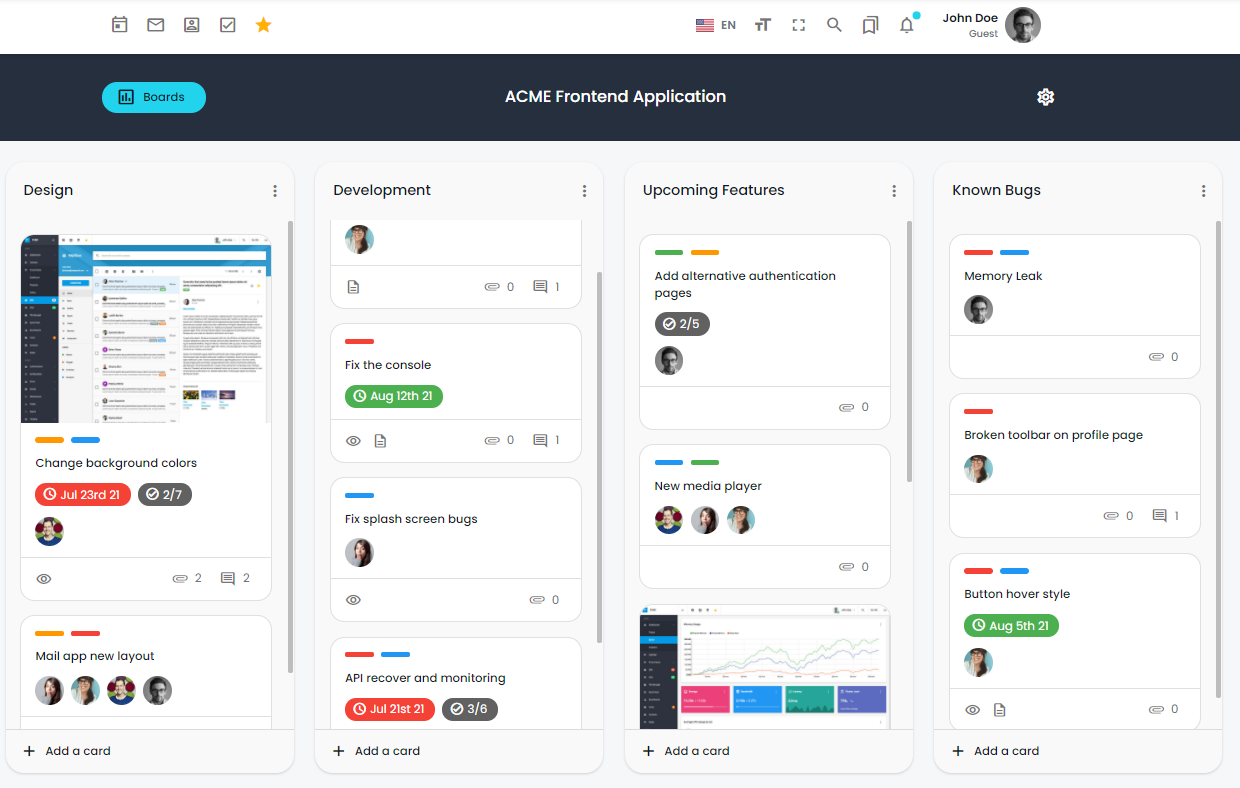
1. Boards



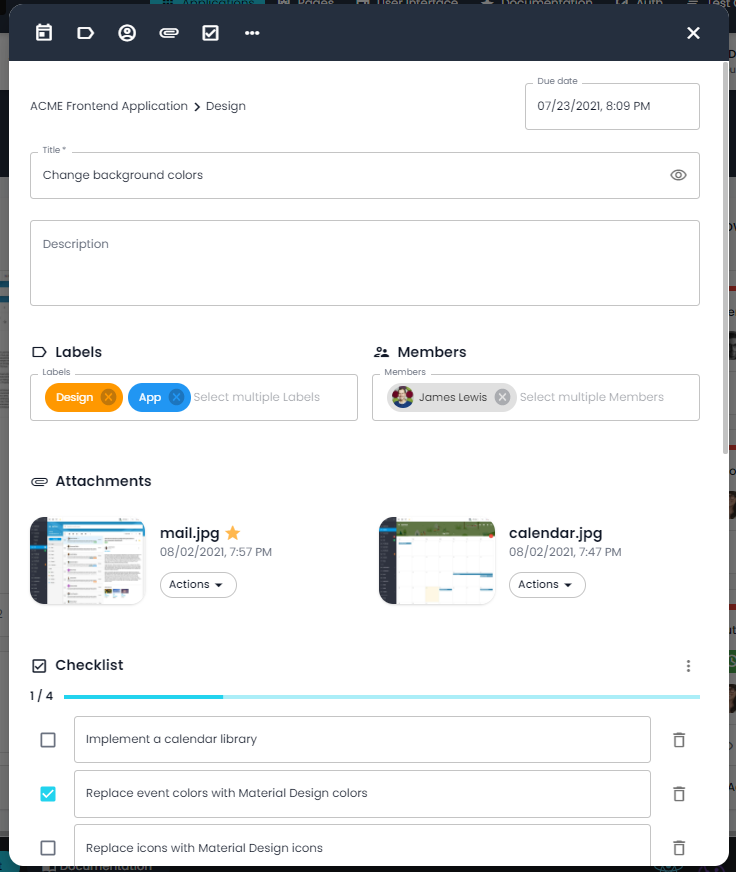
**Purpose**: Organize work like sticky notes and track tasks through every stage. Boards help make your work processes more transparent.

Containts below features:

* Search Board: user can search for their board
* Follow Board: user can watch/delete all their followed boards
* Boards List: user can watch/delete all board they created and assigned
* Add Board: user can create their new board
* When click board, user will be lead to a single board working area:
  + Board: User can edit name, background, add to followed boards
  + List: User can create/edit/delete working list and create card
  + Card: User can edit/delete working card
  + Add Member: User can add more member to this board

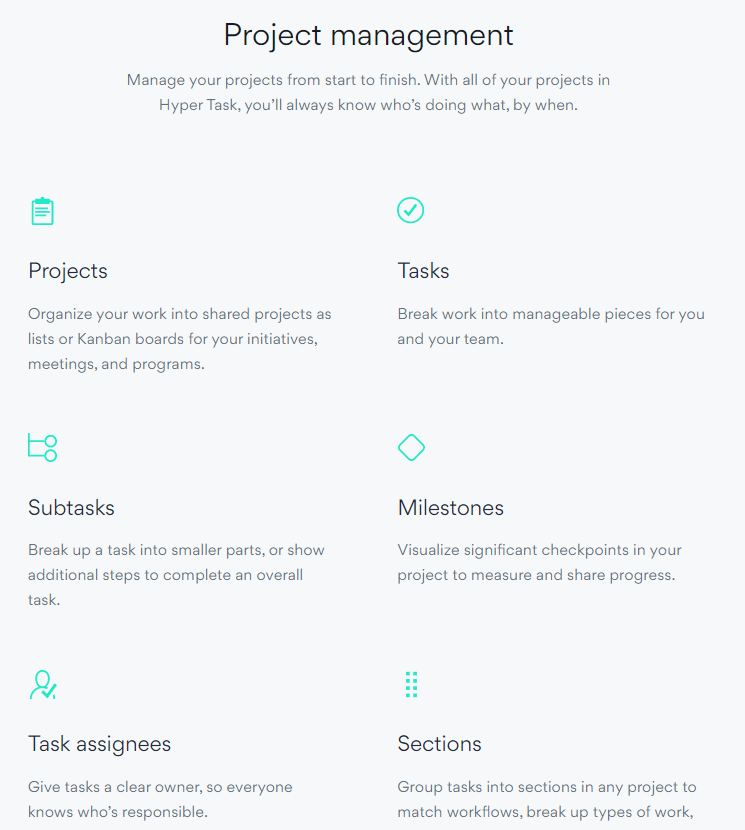
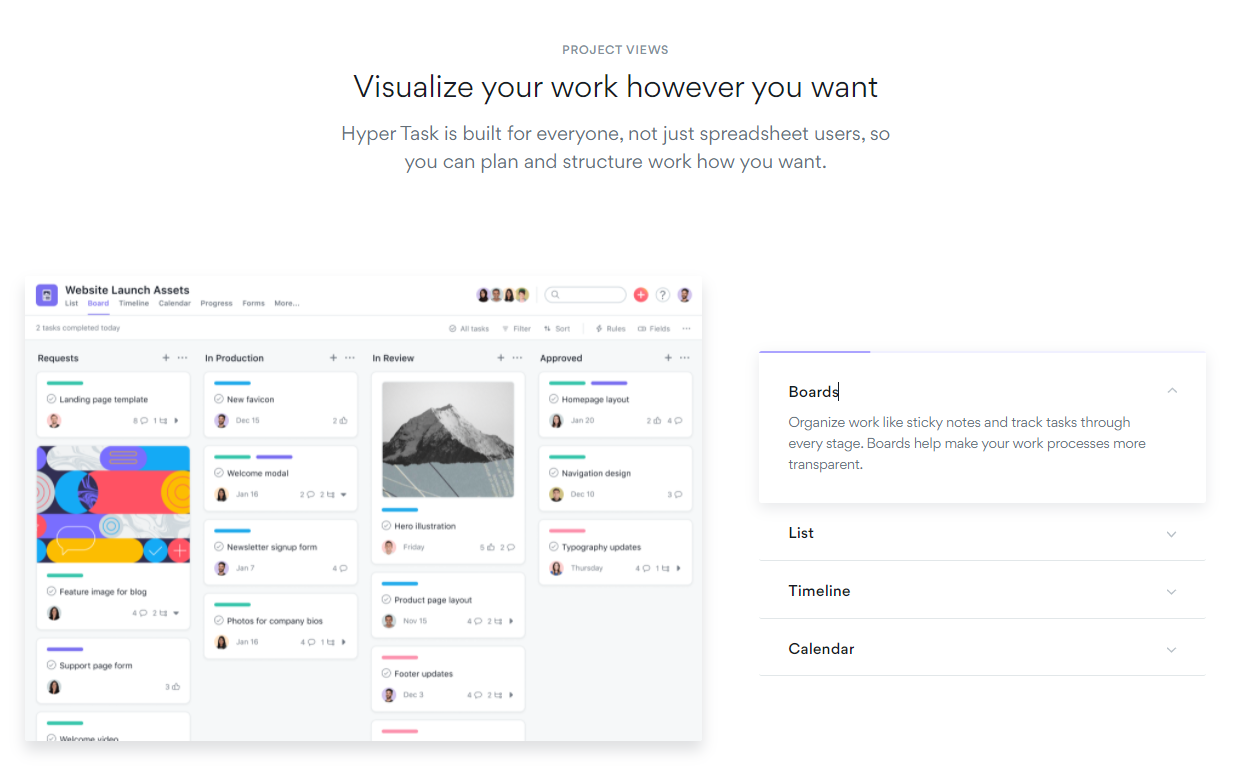
****

**From page**: dashboard – all boards – board details

**To page**: All the links in header-menu and footer-menu link to other pages, home page.

**Description**: This is a dynamic page, get data in the Board and user page.

1. Features

****

**Purpose**: see product details

**From page**: Home - product - product details

**To page**: All links in header-menu and footer-menu link to other pages.

**Description**: This is a dynamic page, data is taken from the News table in the database.

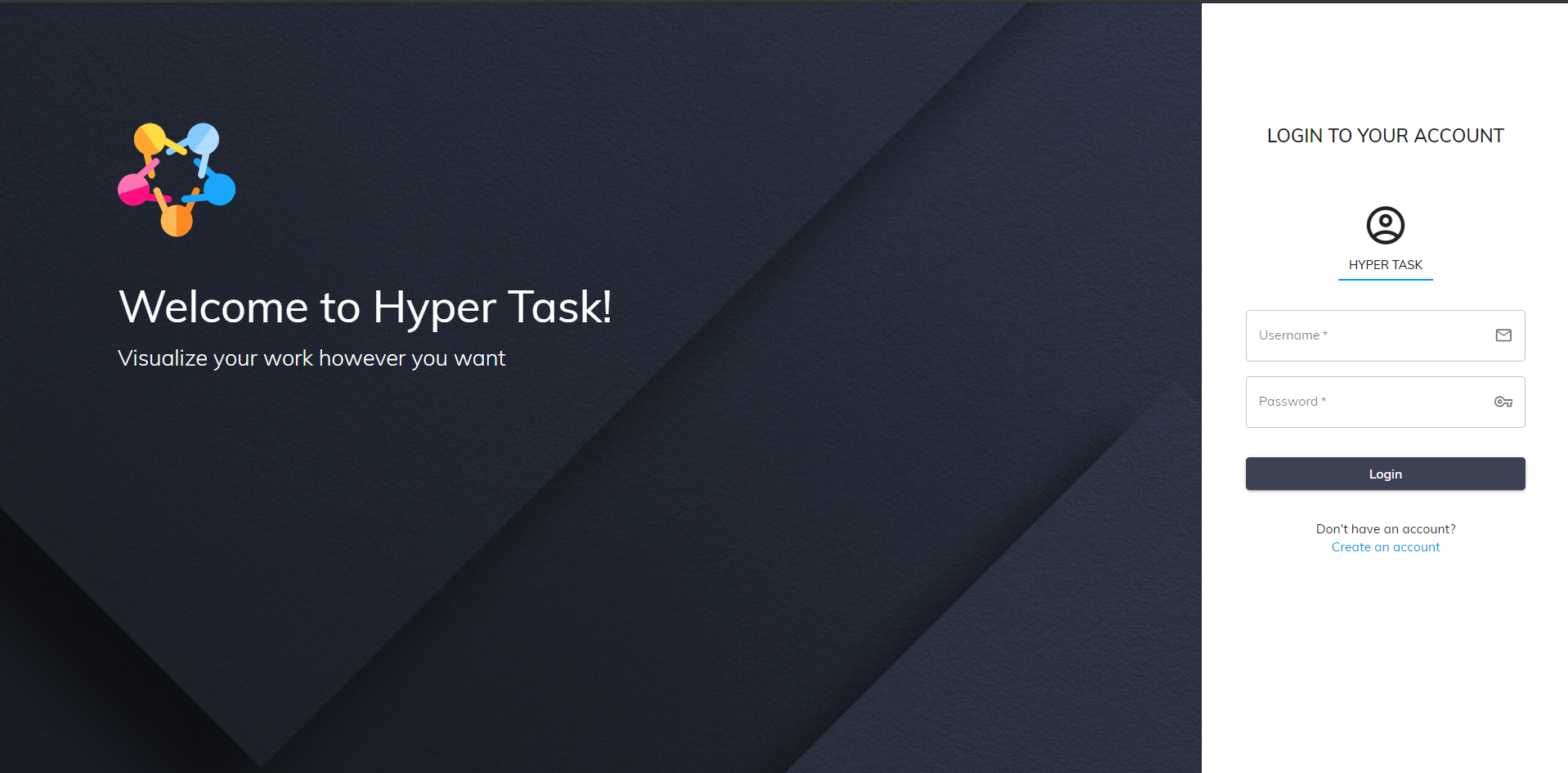
1. Login

**Purpose:** for customers to log in and use the website's functions

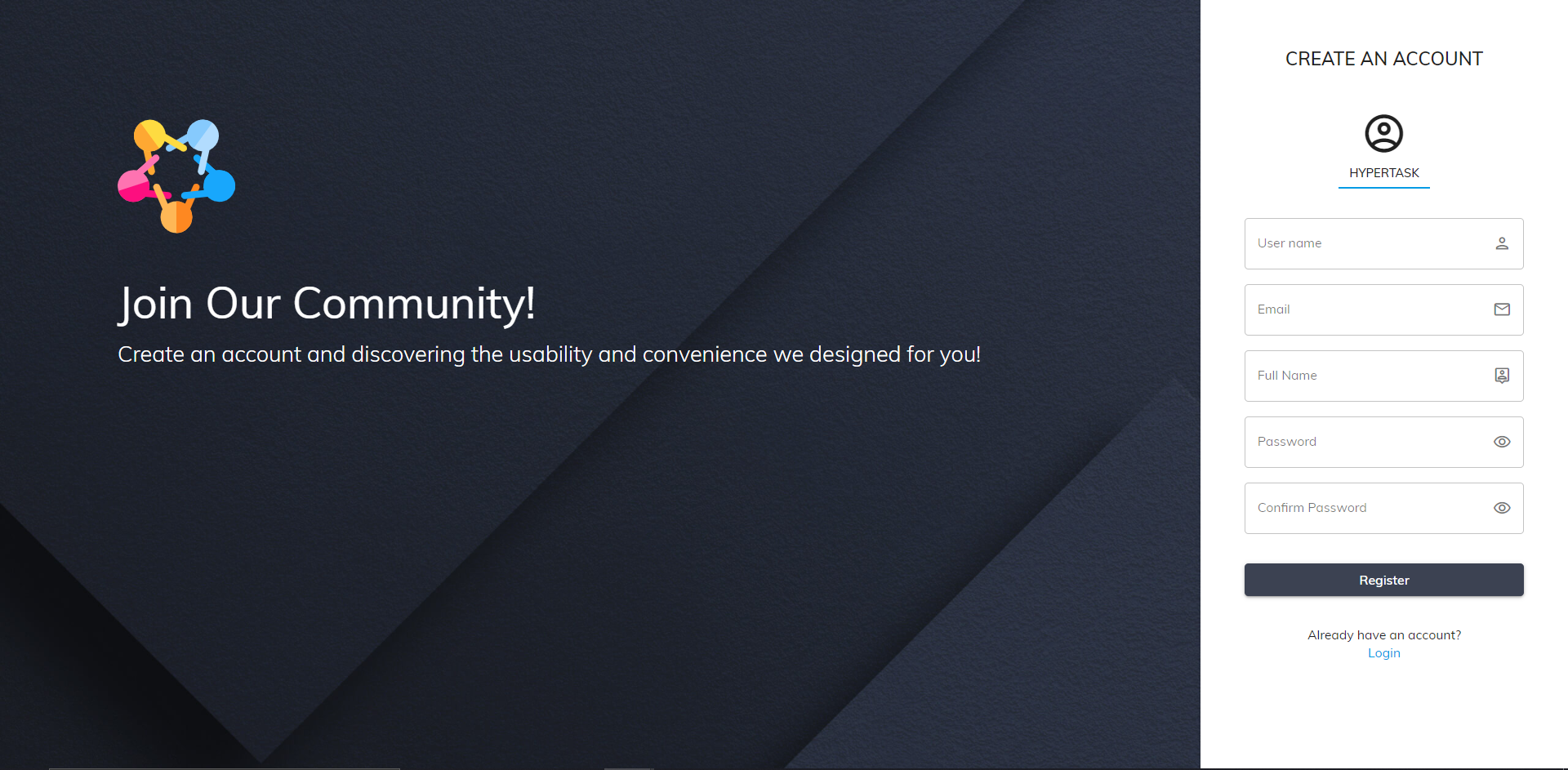
**From page:** The first page when customers enter the website without an account

**To page:** After logging in, the customer will be redirected to the dashboard

**Description:** This is a dynamic page that gets data with 2 types of data returned from this page. If the data is a user, the account can go to the management pages and view the user's information.



1. Register page



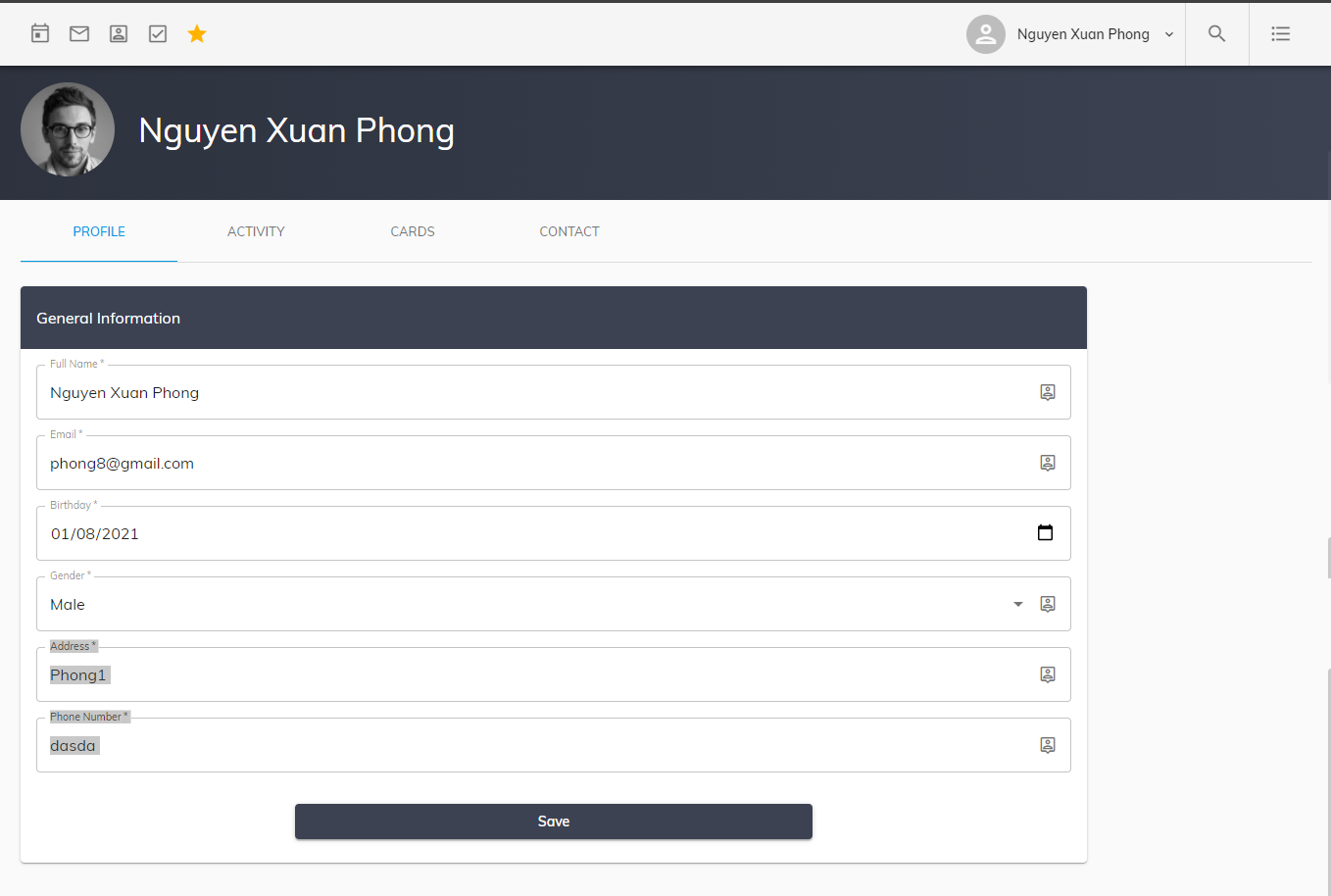
**Purpose:** User can register an account from this site

**From page:** Login -> Register

**Go page:** Login

**Description:** This is a dynamic page where the data will be saved in the database of the account name, password and some other information. After the customer enters the account and registers successfully, it will be transferred to the login page to proceed. Log in to the website system

1. Profile page



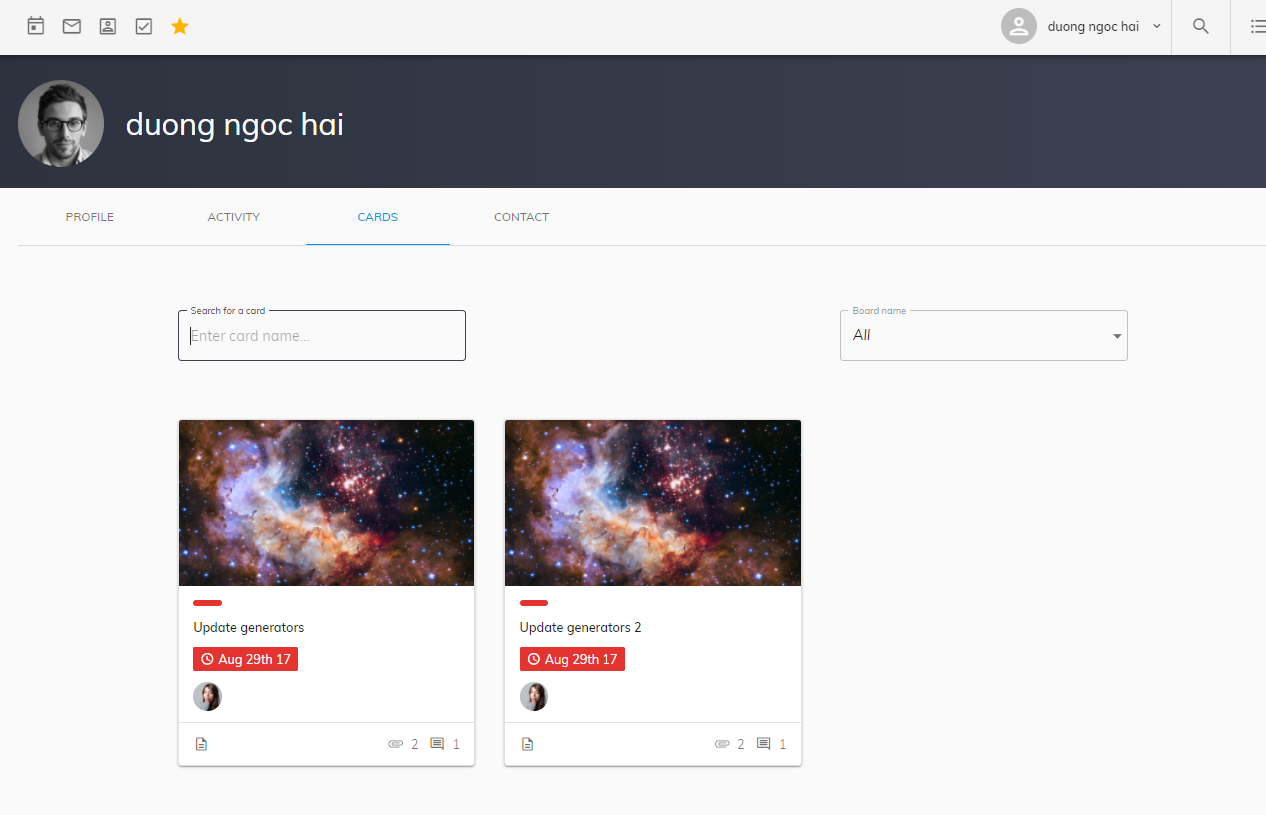
**Purpose**: to show customers account information, and edit their account information.

**From page:** Dashboard - My Profile - Profile Page.

**To page:** Activity Page, Cards Page, Contact Page.

**Description:** This is a dynamic page that takes data from the user table to pour into the table, user can change and update the data in the user table.

1. User’s card page



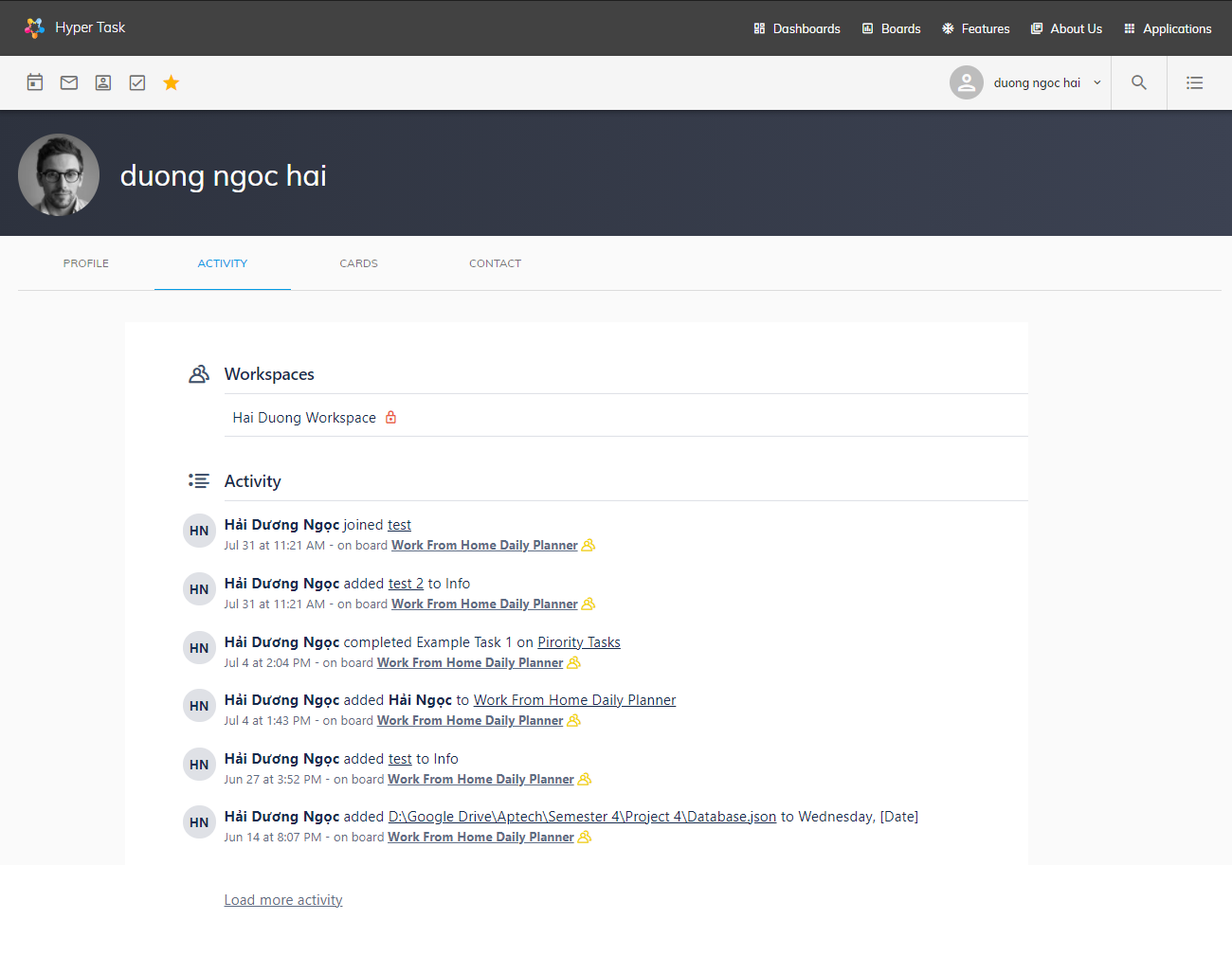
**Purpose**: Display a list of user’s card, these cards may be created by them or assigned to them, so they can track the status of them., quickly view and edit, click on the card to move to the corresponding board.

**From page**: Profile – Cards

**To page**: All links in header-menu and footer-menu link to other pages.

**Description**: This is a dynamic page, get data in the Board and user page.

1. User’s activity page



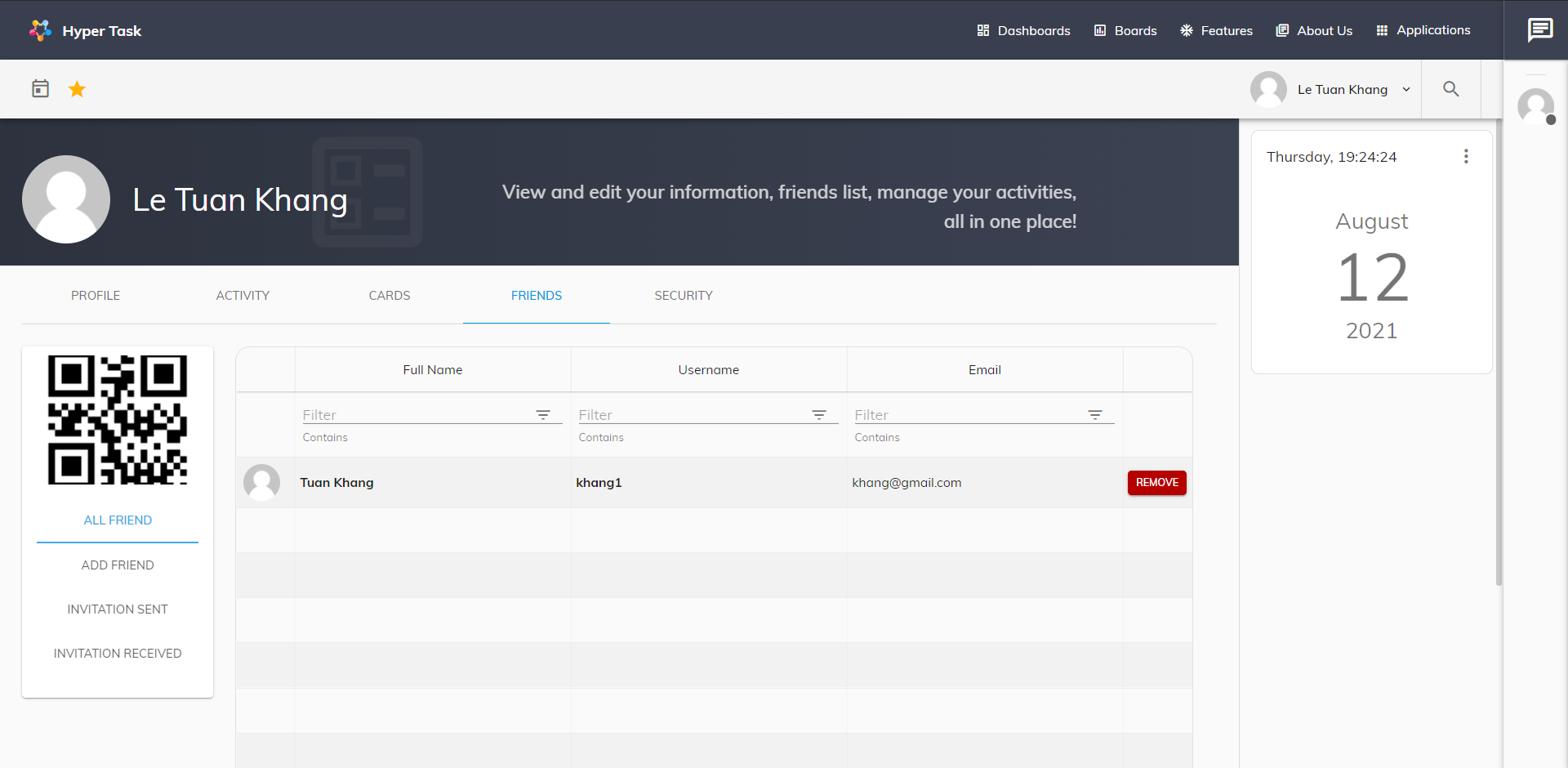
**Purpose**: This page displays a history of user activities on their boards, include many activities such as joining boards, adding, editing, deleting board data, ….

**From page**: Profile – Cards

**To page**: All links in header-menu and footer-menu link to other pages.

**Description**: This is a dynamic page, get data in the Board and user page.

1. Friend Page



**Purpose:** User can view list of they friend, view friend request or accept friend request from this page

**From page:** All Page – Profile - Friend

**Go to page:** All page

**Description:** This is a dynamic page, this page will get the friends list of friends who have agreed to be friends, sent invitations and pending invitations that can do friend search, delete friends, accept and send friend request on this page.

# Task Sheet Review 3

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Members Group** | | **Date Prepareation Of Activity Plan** | | | | |
| **No** | **Name** | **Description Task** | | **Start Date** | **End Date** | **Status** |
| 1 | Dương Ngọc Hải | FE: dashboard, contact us, list (board), user activity, user’s card.  BE: board controller, card controller, file upload controller, user activity controller, user card controller. | | 20/07/2021 | 02/08/2021 | Completed |
| 2 | Nguyễn Hiền Long | FE: board, list, card, board activity.  BE: board controller, list controller, card controller, board activity controller. | | 20/07/2021 | 02/08/2021 | Completed |
| 3 | Nguyễn Xuân Phong | FE: user profile, calendar, card (board).  BE: edit user information controller, calendar controller, card controller. | | 20/07/2021 | 02/08/2021 | Completed |
| 4 | Lê Tuấn Khang | FE: about us, login/register, user’s contact.  BE: login/register controller, add contact controller | | 20/07/2021 | 02/08/2021 | Completed |
|  | | | | | | |
| **Teacher** | | | **Team Leader** | | | |
|  | | |  | | | |
| **Nguyễn Ngô Phước** | | | **Dương Ngọc Hải** | | | |