THOMAS MORE

KMITL

Project plan

Bangkok, Thailand

Jarne Dirken
Kobe Vandendijck
Sohaib Ibenhajene

2023 - 2024

Table of contents

1. Company

KMITL or King Mongkut's Institute of Technology Ladkrabang is a university located in Ladkrabang, Bangkok, Thailand.

There are two other universities called the King Mongkut's Institute of Technology Thonburi, and King Mongkut's Institute of Technology North Bangkok. This organization is dedicated to the exploration, analysis, enhancement, and provision of services in technology, science, and technical education. Its mission also encompasses the preservation and promotion of the nation's art and culture. The school counts more than 20,000 students in all major and educational levels. (kmitl, n.d.)

KMITL comprises 11 distinct schools, which are:

1. School of Engineering
2. School of Architecture, Art, and Design
3. School of Science
4. School of Agricultural Technology
5. School of Industrial Education and Technology
6. School of Food Industry
7. School of Information Technology
8. KMITL Business School
9. School of Liberal Arts
10. Faculty of Medicine
11. School of Dentistry

2. Problem

KMITL has a lot of hardware components that students can borrow if they want to. That’s why the school decided to make an inventory managing system. This way students can borrow items, return them and everything can be monitored by the application so no components should get lost.

However, the existing system, while effective in its primary goal, exhibits several limitations that require attention. Our task is to address these shortcomings to enhance the system's functionality and user experience. The identified issues include:

- The problem is that the current system doesn’t have a good authentication system. Users can just create an account with whatever student number, email, name, etc…
- The absence of an approval process for borrowing items enables individuals to potentially borrow an entire stock without oversight, leading to potential abuse and inventory management challenges.
- The system's design is not fully optimized for mobile devices.
- The organization of the inventory table falls short of expectations, making it difficult for users to navigate and locate specific items efficiently.
- The overall design of the system is overly simplistic, which, while minimalistic, may not engage users effectively or provide an intuitive user interface.
- User-friendliness is compromised, indicating that navigating the system and performing actions like borrowing or returning items could be more intuitive and less cumbersome for users.

Our involvement will focus on developing solutions to these issues, aiming to create a more secure, accessible, and user-friendly system that meets the needs of both the institution and its students. By addressing these challenges, we will ensure that the inventory management system not only functions effectively but also enhance the user experience for all stakeholders involved.

3. Objective

Dealing with borrowing components at university can really be a headache for students. University staff don't enjoy chasing after items they've lent to students either. We're spending our internship trying to fix this problem.

Our plan is to make a web application specifically for KMITL students and staff. It's going to be very straightforward and easy for everyone to use, but it will also have plenty of useful features. This web app is all about making sure students and staff can work together smoothly, so nobody must wonder where something is or who has borrowed it.

For building this app, we're using Next.js as our main technology and PostgreSQL for handling all the data. We want to make this app a lot better than the current one. Some new things we're adding include a system where staff can approve loans easily and a feature for checking items in and out with QR codes. The webserver should run on a local server in KMITL itself that we have access to. We have to keep in mind that the system will be used by a minimum of 600 students so security, performance and scalability will be key factors in our project.

With these updates, borrowing components at the university will become a whole lot easier and more organized.

4. Business case

4.1. Added value

Our project will offer significant value propositions with broad-reaching benefits. We stand to enhance the current borrowing system with the help of new technology and features.

With the help of our project, we reduce the risk of lost or stolen items through better tracking, monitoring, and authentication. Furthermore, we will be facilitating a smoother borrowing experience, saving time and reducing frustration for both students and staff. We mustn't forget to make our solution scalable that meets the growing needs of the KMITL community.

4.2. Stake holders

KMITL Staff: Our primary stakeholder is the staff of KMITL. They will benefit from improved inventory management, better clarity and role management.

KMITL Students: Students will benefit from this system by having a better user experience while borrowing or returning an item.

5. Project timeline

5.1. The steps that are going to be taken

Our project consists of two phases. The initialization phase and the realization phase.

5.1.1. Initialization phase

In this phase we will be focusing on our project plan (this document). This phase consists of research only. These are the steps we are going to take:

1. Talk to someone who uses or knows the current system and ask for an opinion.
2. Inspect the current system and try to look for improvements.
3. Research the tools we are going to use for the new system.
4. Make the screens in Figma.

The initialization phase should only last 3 weeks. **4/03 – 24/03**

9	04/3	Initiation phase	Internship 1
10	11/3		Internship 2: kick-off meeting
11	18/3		Internship 3

5.1.2. Realization phase

In this phase of our project, we will be developing a new system. We have done all the research so now it's time to put it into practice. These are the steps we are going to take:

1. Make a basic front-end and back-end with dummy data and make sure they can communicate with each other with the help of an API.
2. Improve the application so it looks like our Figma board with all the requirements.
3. Make everything work as needed, remove bugs and put in some real data.

The realization phase should last all the other weeks (10): **25/03 – 24/05**

12	25/3	Realization phase	Internship 4: 1st meeting at school
13	01/4		Internship 5
14	08/4		Internship 6: intermediate internship evaluation
15	15/4		Internship 7
16	22/4		Internship 8: 2nd meeting at school
17	29/4		Internship 9
18	06/5		Internship 10
19	13/5		Internship 11
20	20/5		Internship 12: submit internship evidence documents for review

6. Communication and responsibilities

Our supervisors will check up on us every week. If they don’t have time to check up on us, they will send over a senior student who uses the current system and he will look at what we have done and report everything back to the supervisors.

Because we will be working on a web application, we can host the website and present it live. They can follow up our work remotely by looking at the website.

Kobe will be doing all the infrastructure and security of this website. He will set up a self hosted GitLab server and secure the pipeline with SAST,DAST and WAF. For hosting the website he will use the things already in place from the old website. He will reuse the SSL/TLS certificates and the API calls will be secured with API keys.

Because we will be working with two full stack developers on this project we are going to divide the work. Jarne will mainly work on the student and the login pages. Sohaib will mainly work on the supervisor pages. We’ll both be developing the admin pages as well as any extra features that should be included such as alerting, shopping cart, …

7. Project scope

Here our MoSCoW of the project.

The MoSCoW method is a prioritization tool in project management, dividing tasks into:

- Must have (M): Essential for success.
- Should have (S): Important but not critical.
- Could have (C): Nice to have, if possible.
- Won't have (W): Not included in this phase.

This helps focus on what's crucial for the project's outcome.

Must have:
- Web application
- Managing inventory
- Borrowing items
- Returning items
- Approval based borrowing & returning
- Creating account
- QR code generating
- Transaction history
- Repair history
- Basic infrastructure
- TLS/SSL certificates
- API keys
- Domain name

Should have:
- Item's transaction history
- User transaction history
- Responsive to fit mobile screens
- Account summary
- Add push notification 4 conditions: loan, return, almost expired(StSV), expired (StSV)
- Filter on specific attributes
- Urgent borrowing process
- Excel import
- Improve repair flow- icon addition: expired
- Backups (snapshots vSphere)
- Advanced borrow system
- Multi loan / delete

Could have:

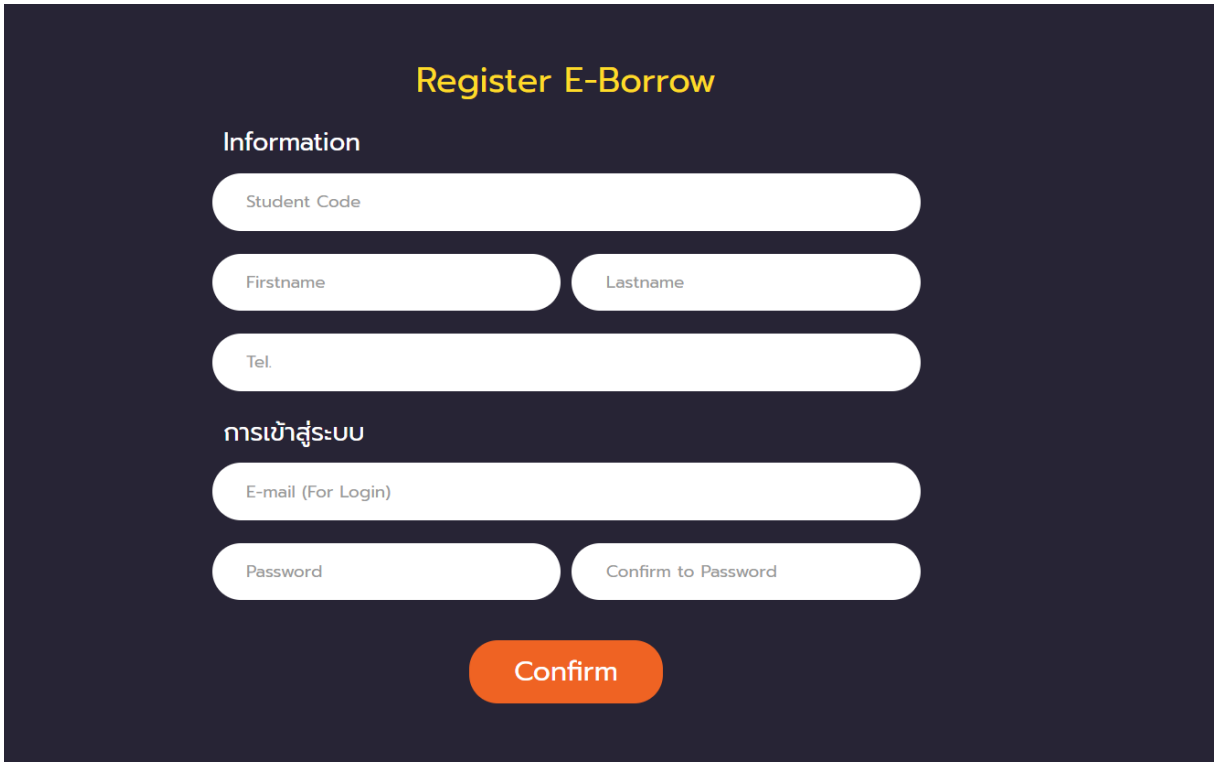
- Export item history
- Export user history
- Export repair history
- Advance booking features (reserve item)
- Add message when returning borrow
- Supervisor / admin change borrow status
- Add message on repair item
  Repair history in item history
- Log of all activities (admin only)
- Low stock option alert
- Automatic number appending

Won’t have:

- Mobile application
- Any AI driven features
- Any analytics
- Importing all old data
- Placing QR code on all hardware components

8. Inspection of the current application

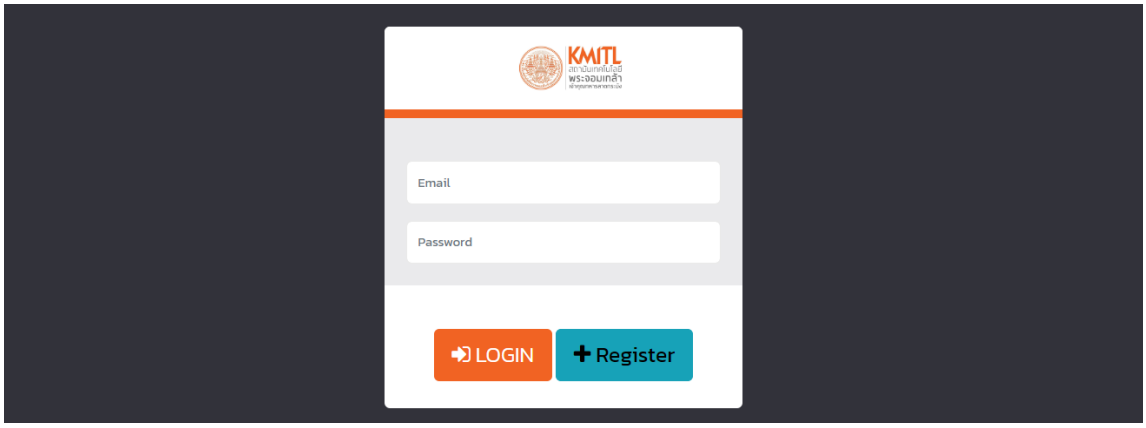
8.1. Register form



When we first saw the application, we had to register to use the application. The first thing we noticed is that we can just enter anything anywhere. Meaning if we want to fill in "1" everywhere, we can. There is no validation, which is not good.

Also, we can see that there is no "back" button. If we accidentally click on "register", there is no way for the user to go back.

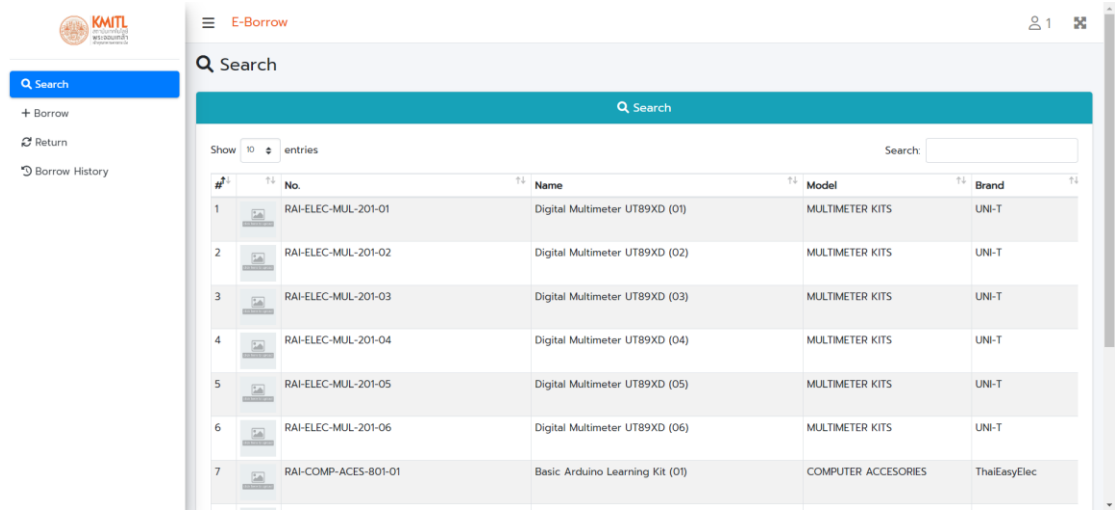
8.2. Login form



When filling in nothing and pressing “login” you don’t see any error messages. This is not user friendly. Only when you type something and try to log in, you can see an error message (in Thai, so not in the language of the browser).

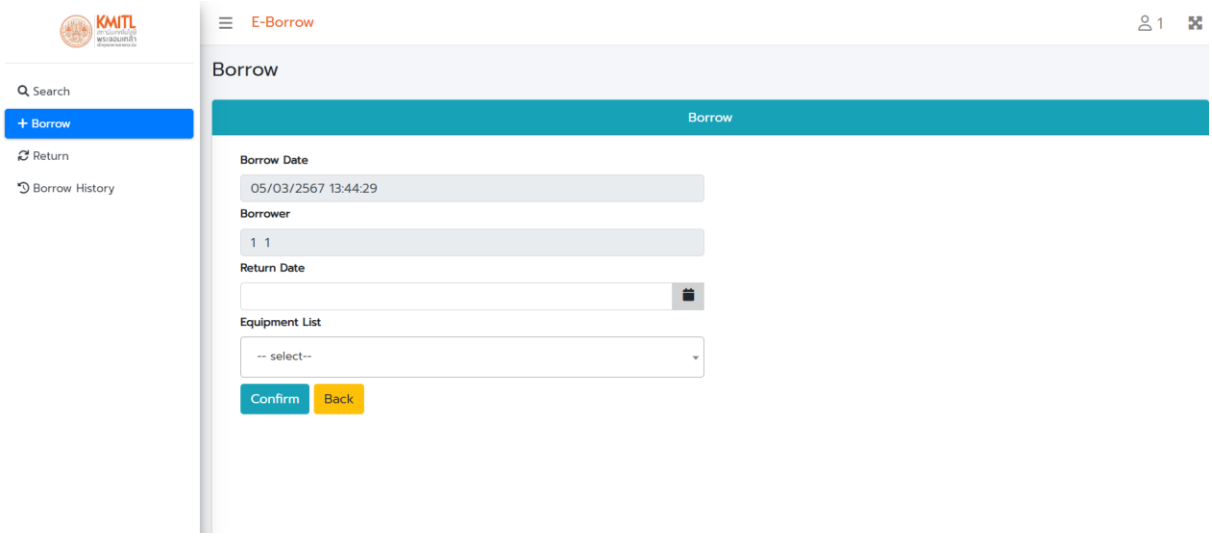
Additionally, we are concerned that this system, with its custom registration page storing data in a local database without adequate verification, coupled with a login page that fails to validate input values properly, could pose a significant security risk.

8.3. First impression



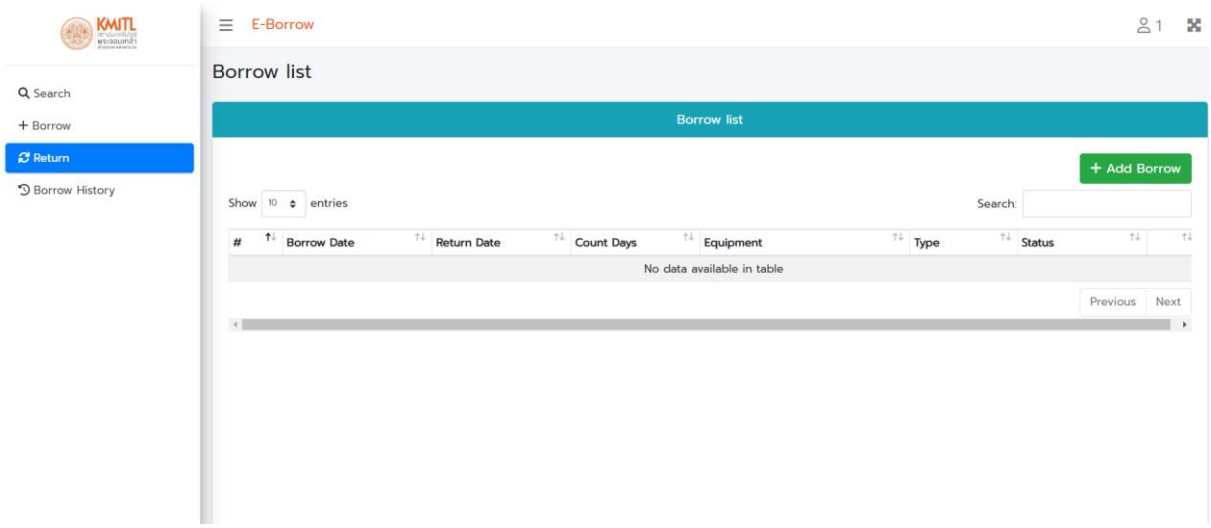
That is what the landing page looks like when you first login, there is a lot going on. On the left side is the navigation bar with all the pages, and the rest of the page is a view of the database with all the available items. The navigation bar looks very good and user friendly, it is simple and easy to use and folds in if you want it to. The main part also has a search function, where you can search for anything by name, no brand or model. We noticed that if there are multiple items that are the same, they all show up on this screen. The last thing we noticed is that the database is not shown in full on the page, on the bottom there is a slider for showing the rest. These last 2 things make the first page a bit overwhelming and confusing.

8.4. Borrow page



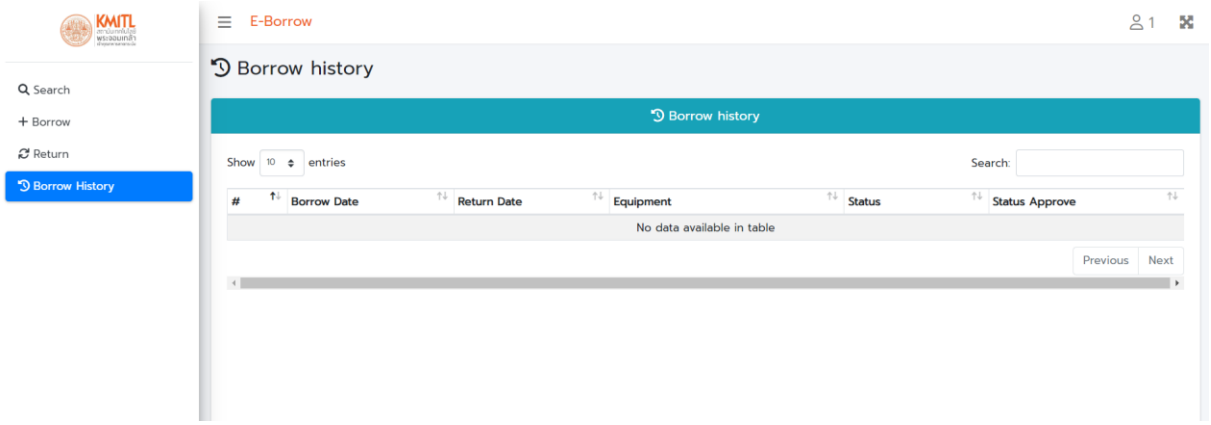
The borrow page still has the clean navigation bar. The rest of the page looks good as well, and it is easy to use. The best part of the page is that it fills in the date automatically, so you don’t have to go look for it. There is also a button that sends you back to the home page. The only thing on this page that bothered us is that the search bar gives you again all the items with the same name, and that makes it a bit chaotic.

8.5. Return page



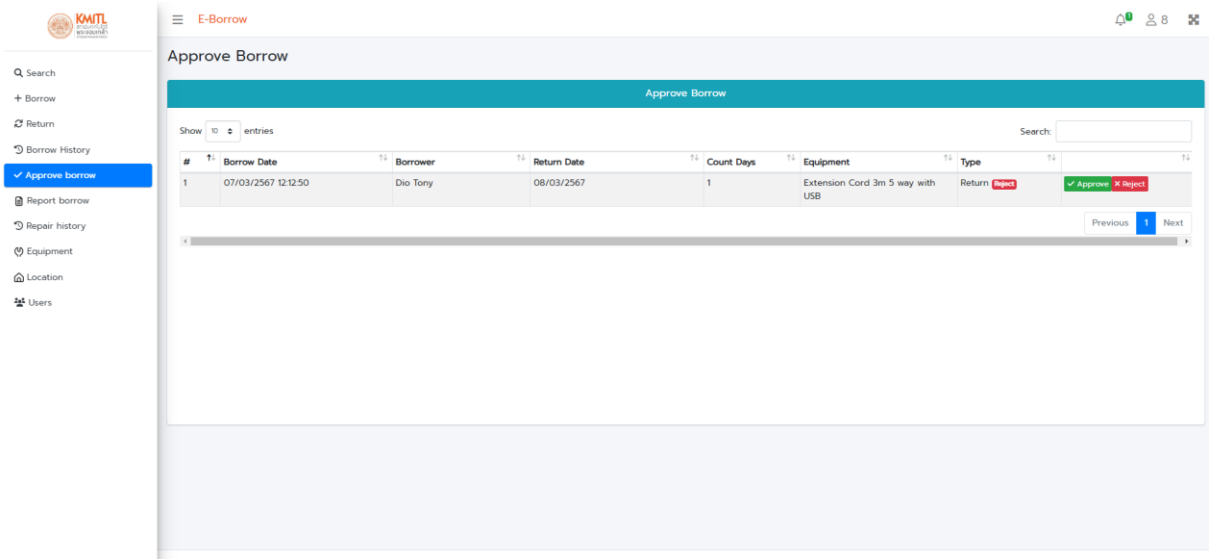
On this page is shown what you have borrowed from the university that you have not brought back yet. We like this concept very much, as well as the add borrow button that send you to the borrow page.

8.6. Borrowing history page



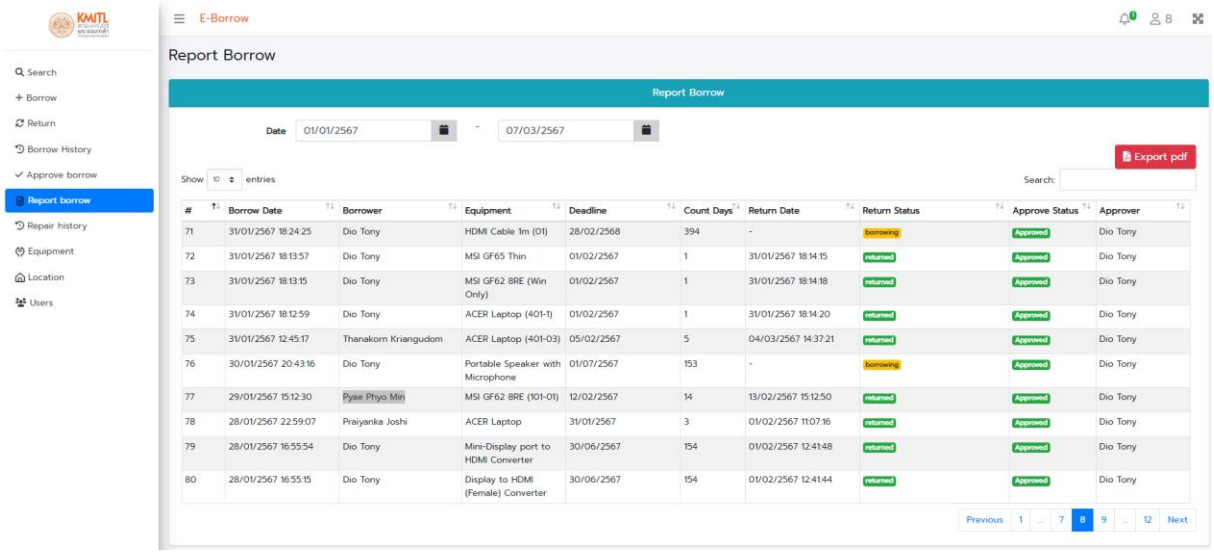
The borrow history page shows you everything you have borrowed in the past and have brought back. This is also a nice addition to the inventory system.

8.7. Approve borrow (admin)



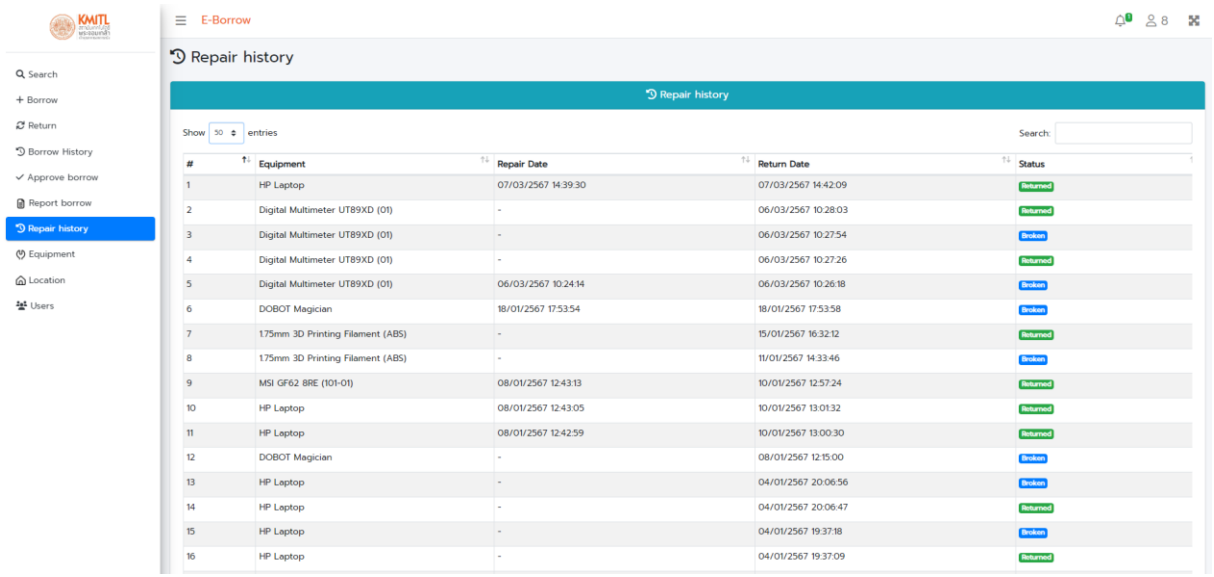
Now let’s look at the website from an administrator perspective. The first pages stay the same as before, but certain pages have been added. Starting with Approve borrow, on this page there is a list shown of all the request to borrow something as well as all the requests to return something. This would be quite chaotic if the system was in full use and there were more requests. For each request there is a button to approve or reject the request and each button triggers a notification pop-up.

8.8. Report borrow (admin)



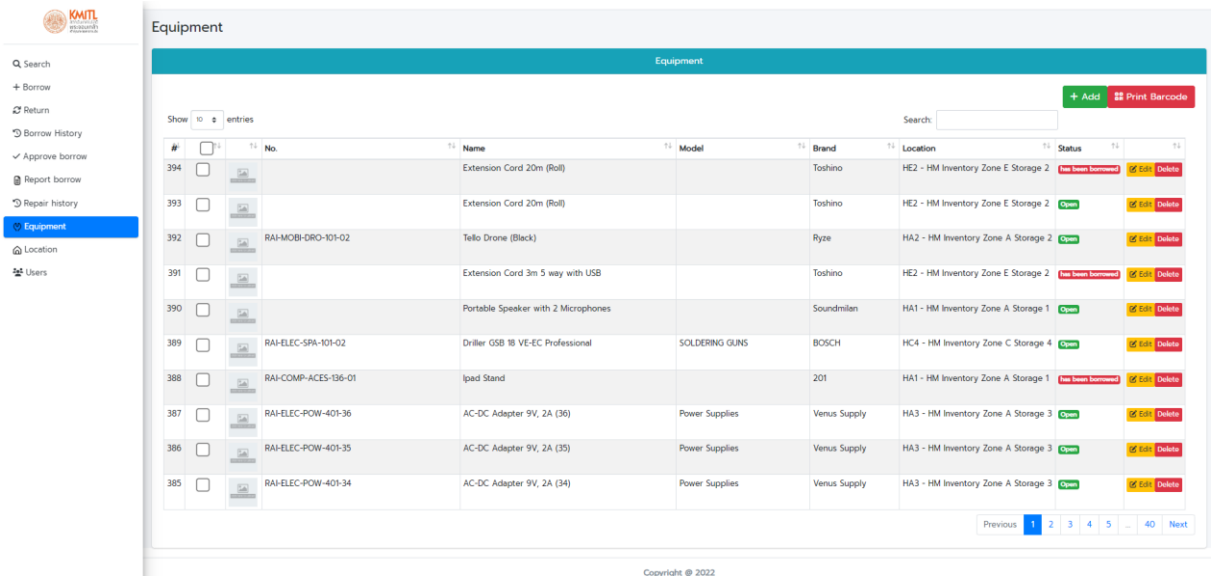
This page is very good, you can filter on dates, so that you don't have too many entry's. You can also export as pdf with a button for printing out as well as a search bar and the sorting is handy on this kind of page.

8.9. Repair history (admin)



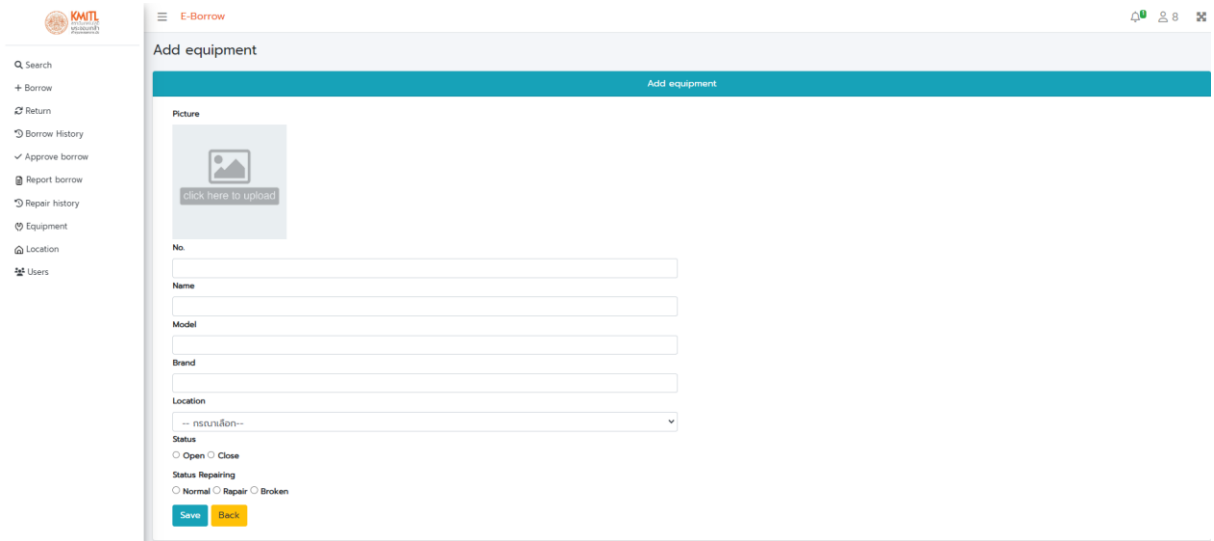
This is also quite a good and clear page, the few columns make it an easy to understand page without confusion. Maybe adding a date filtering would be good if the application was bigger and more items were being used.

8.10. Equipment (admin)



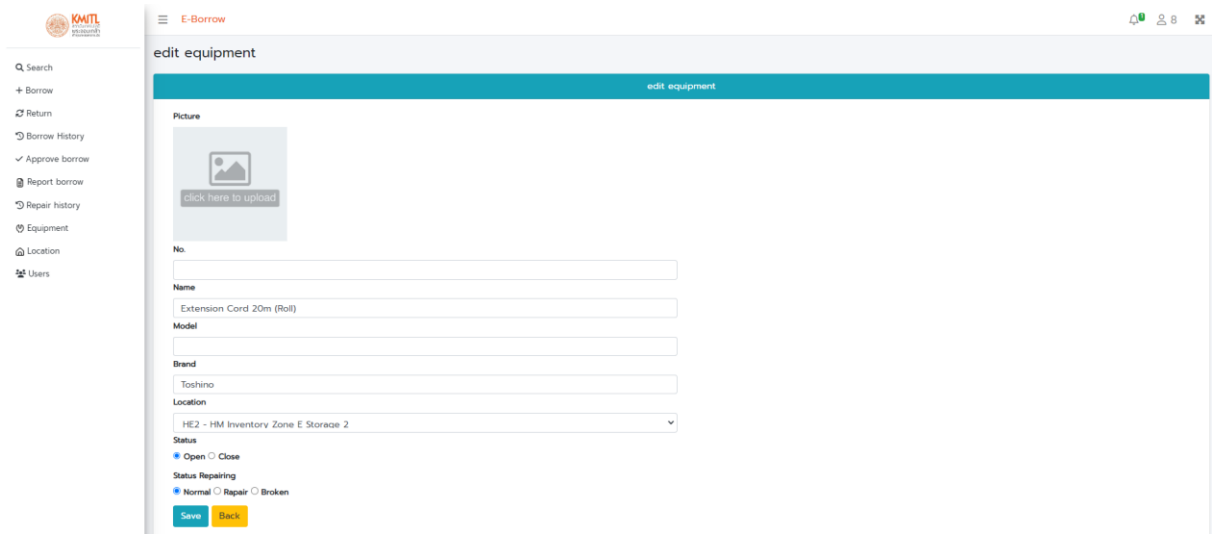
This page displays all the equipment that can be borrowed from the university. This also gives the administrator some options, you can add, edit or delete the items. Different to the search and borrow pages where every single item is shown, what we do not like. On this page it is actually really helpful to have. There is also a button on the screen that allows you to print out the QR-codes.

8.10.1. Add equipment



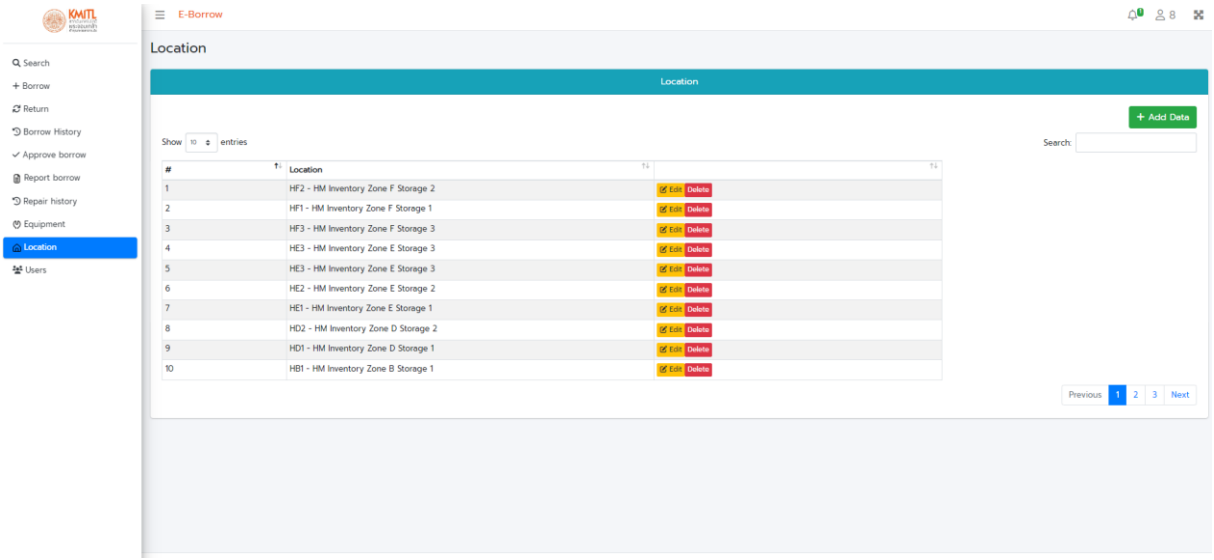
The “add” button on the equipment page leads you to another page. On this page you can choose to add an item to the list. Here you can enter its basic information, as well as a picture. The bad thing we noticed here is that the only required field is the location, this enables people to add empty items to the list.

8.10.2. Edit equipment



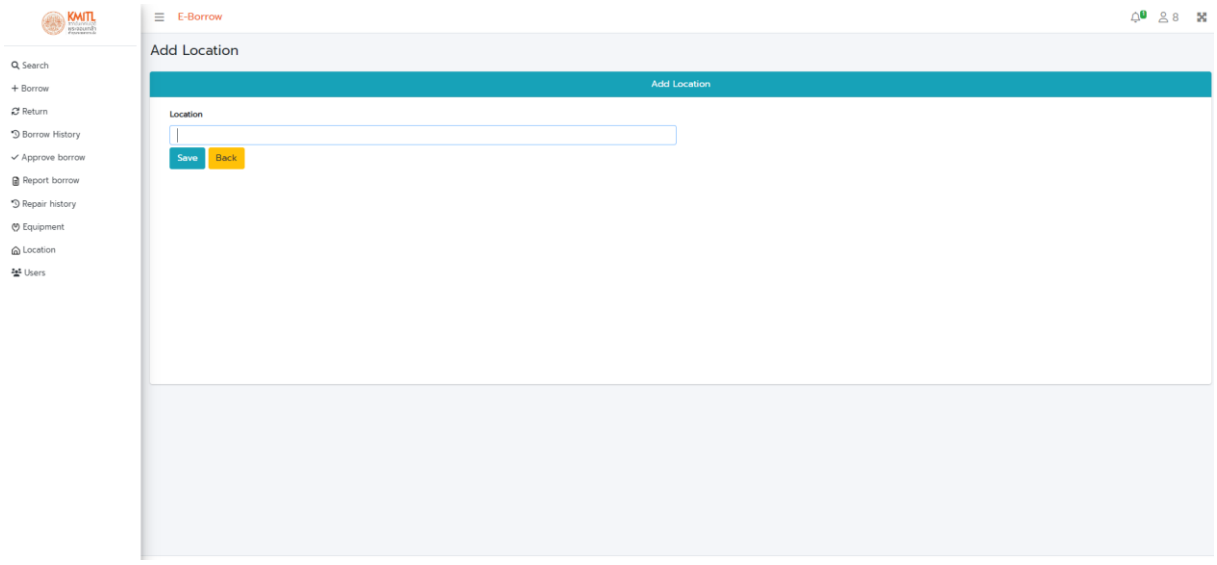
If you push the edit button you go back to the add equipment page, with the only difference being that the known information is filled out in the form and the title is different.

8.11. Location (admin)



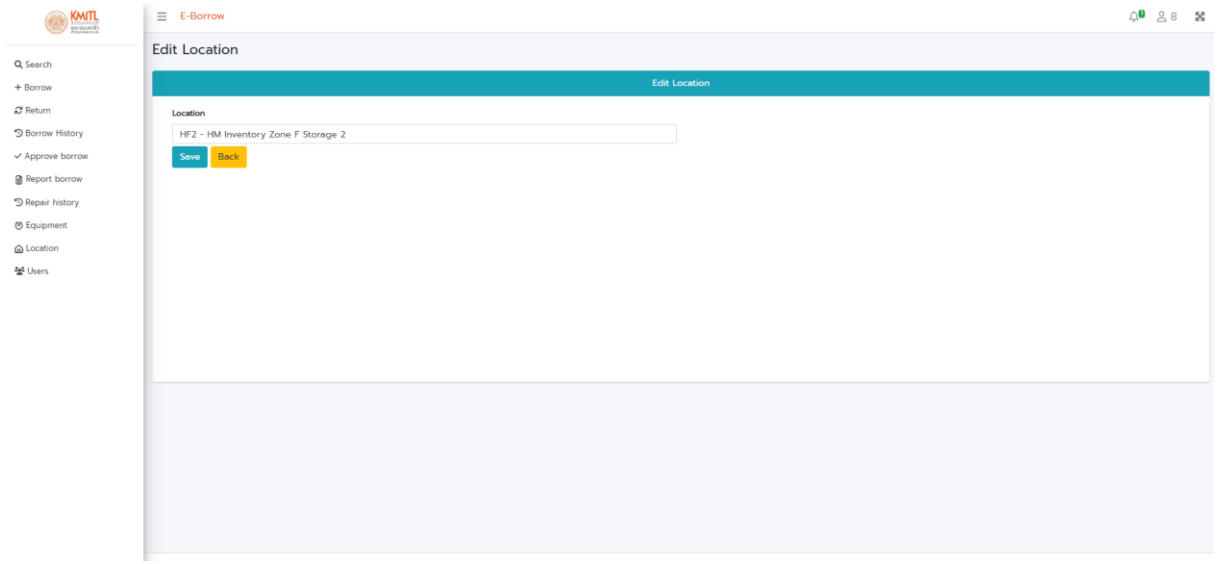
This page is again quite good, it shows all the locations where there is storage for equipment. On this page you can add, edit or delete locations. You can again search for one as well. On this page we noticed that if you have less entry's than the amount specified on the top, the button for going to another page jumps with it to the top, staying just underneath the last piece of data.

8.11.1. Add location



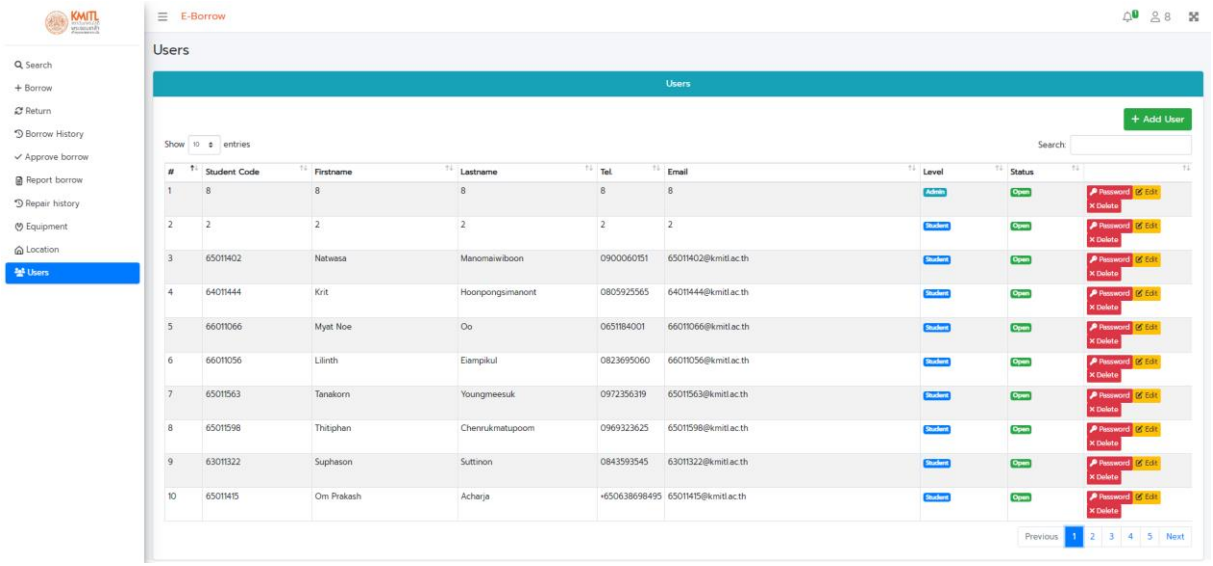
On this page you can enter a new location, otherwise it is empty. There is no basic format in which a location can be entered. This could be done in order to make sure that no one can just enter anything.

8.11.2. Edit location



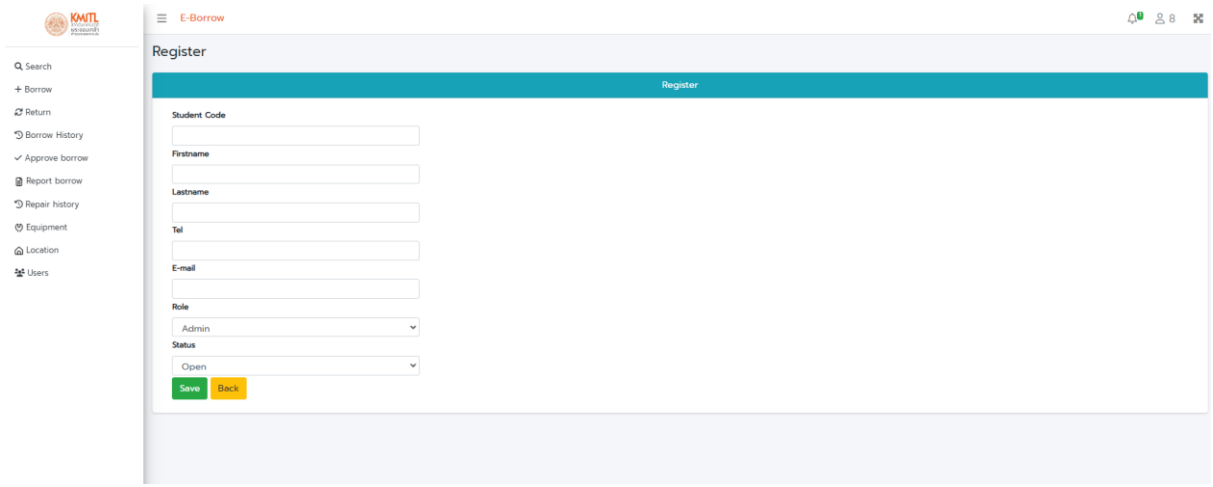
On this page you can edit the location data, it is again empty other than the one form element.

8.12. Users (admin)



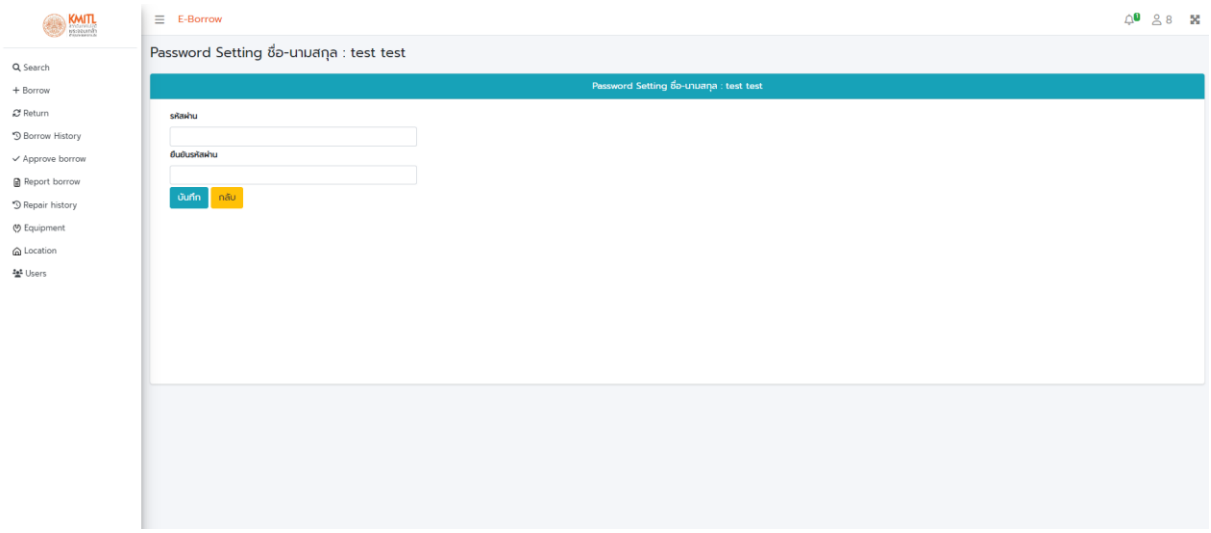
On this page we can see all the registered users of our web application, you can see basic information as well as level clearance and status. You also have a couple of buttons that you can use: add user, password, edit and delete.

8.12.1. Add user/register page



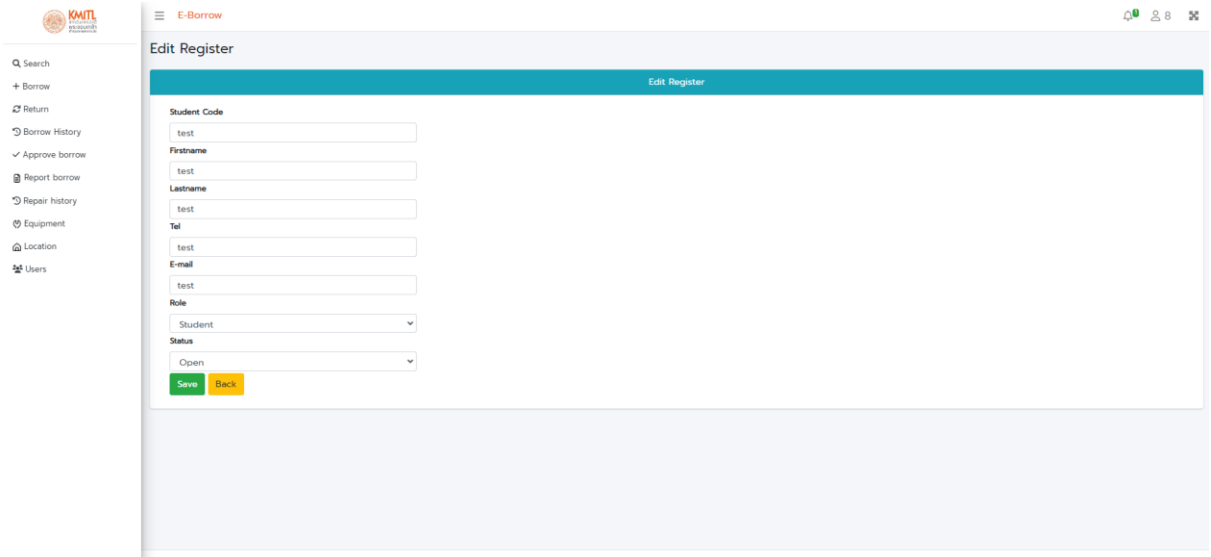
If you click on add user, you are sent to a page called register. On this page you can add basic information of a user, as well as set the role they have and the status of the account. You can not enter a password on this page so that is some extra work you will have to put in to create a user.

8.12.2. Password setting



This page only contains a form that lets you add and confirm a password of the user you have selected. You can change the password of a known user as well, it does not matter what role they have. The users don't even get a message to let them know their password has been changed. This could be a severe security breach if someone ever got in. You could even change an admin password if you wanted to.

8.12.3. Edit register



This page allows you to edit the basic information of a user, you can take away/give admin rights to/from the users. You can also change the status of an account to closed.

8.13. General

To give a general overview of the current application. It looks good and is very fast. There are things that should be fixed like a user profile page. The f11 button is kind off useless. The sidebar isn’t properly aligned. The table is too big and doesn’t look good on mobile. There is no option to filter on multiple criteria.

Also, we think it might be best to combine the return and history page together into one page. This could make it easier to understand.

The security also lacks some things that we will need to change on our own application. And we think that merging some of the extra pages (i.e. add, edit) would be beneficial as well.

Overall, a good start but it feels like a half-finished product. We will do our best to implement everything they asked for so that the new system is everything they need, and they can step away from the google forms.

9. Research

9.1. Front-end

Next.js is widely acclaimed for its exceptional capabilities in the development of modern web applications, especially as a frontend framework. Utilizing React, it offers an enhanced experience for building user interfaces, incorporating server-side rendering (SSR), static site generation (SSG), and client-side rendering methods to optimize performance and user experience across the web.

Next.js includes automatic image optimization with the next/image component, which serves optimized images in the formats supported by modern browsers, resizing images on demand. This feature significantly improves loading times and performance by reducing the size of images without compromising quality.

Built-in internationalization (i18n) support in Next.js allows for the creation of multi-language web applications with ease. It provides automatic language detection, URL routing, and efficient loading of localized content, making it simpler to develop global applications. Which would be very helpful here so we can just detect the language of the browser and give the error messages in that language.

9.2. Javascript VS Typescript

For next.js we can choose whether we want to build our app with JavaScript or typescript. So, which one is better? Well, they both have their pros and cons. JavaScript is easier to learn and more forgiving but that also means it’s more prone to errors and really difficult to scale for larger projects. Typescript on the other hand is a superset of JavaScript so all valid JavaScript code will also work in typescript but the extra features that It’s less forgiving so it can catch errors faster.

Ultimately, the best language to use to build Next.js applications is the one that you are most comfortable with and that best meets the needs of your project. If you are new to programming, I recommend starting with JavaScript. Once you have a good understanding of JavaScript, you can then decide if you want to learn TypeScript. In case you like to build production level projects with huge code base without a doubt typescript will be the best option for you. (Salman, 2023)

Next.js is built with TypeScript under the hood, so you get better IntelliSense and type definitions in your editor by default with just JavaScript. But when you couple that with TypeScript, you can get an even better developer experience — including instant feedback when your component expects props, but you didn’t pass any. Well, this is something most IDEs understand even if you don’t use

TypeScript. But as you work with Next.js, you will build many components with optional props, as well as many utility functions which can be enhanced with TypeScript for better bug spotting, documentation, IntelliSense.

You're also able to build with Next's exported types and define your own to build with across your applications. These types help give your code better structure by dictating what your objects, arrays, etc., look like ahead of time. That way, you, your code editor, and any developer after you knows how to reference your code. (Chaudhari, 2023)

So, in the end what does it all comes down to? Are we working on a bigger project, using typescript for more documentation, better bug spotting and better intellisense. That's why for this project we are going to use Typescript.

9.3. Back-end

As for our back-end we should also use Next.js. This is convenient because we don't need to change languages, we can just use Next.js for front- and back end.

Here we will write API-calls to our database to show the data on our website. One of the core features that facilitate Next.js's use as a backend solution is its API routes. These routes allow you to create RESTful APIs directly within your Next.js application. By placing any file under the app/api directory, you can create an endpoint that behaves as an API. This setup simplifies the development process as you can manage both your frontend and backend code in a single project, making it easier to develop, test, and deploy your application.

API routes in Next.js are essentially serverless functions that run on demand, scaling automatically with the number of requests. This serverless approach reduces the overhead of server management, ensuring that your application can scale effortlessly without the need to manage infrastructure. It's particularly beneficial for applications with fluctuating traffic, as it can dynamically allocate resources to meet demand.

Handling authentication in Next.js apps can be managed through API routes, utilizing libraries such as NextAuth.js. This library simplifies the implementation of authentication systems, supporting various authentication providers and strategies, including email, social login, and JWT tokens. Moreover, Next.js apps can be secured using standard web security practices, such as secure headers, HTTPS, and data validation/sanitization to prevent common web vulnerabilities.

9.4. Database

PostgreSQL is a powerful, open-source object-relational database system known for its robustness, scalability, and adherence to SQL standards. It has become the go-to database for a wide range of applications across various industries, from startups to large enterprises, due to its advanced features and reliability. PostgreSQL offers a sophisticated yet practical solution for managing data, regardless of the complexity or volume, making it an ideal choice for both traditional and modern, web-facing applications.

Data integrity and reliability are paramount in PostgreSQL, with features such as atomicity, consistency, isolation, durability (ACID) properties, sophisticated locking mechanisms, and foreign keys ensuring that the database remains consistent and robust under various conditions. It also supports savepoints and point-in-time recovery, enhancing data protection and allowing administrators to restore data to a specific moment in case of an error or system failure. This is very useful when working with a lot of data that changes constantly.

In order to create the database and make a connection with it we are going to use Prisma.io. The purpose is that Prisma simplifies database access by providing an easy-to-use API for querying and managing our database. Prisma is used on the backend to make sure our database is created with schemas and for data fetching. Or as Prisma like to say it:

"Prisma makes working with data easy! It offers a type safe Node.js & TypeScript ORM, global database caching, connection pooling, and real-time database events." (Rauch, n.d.)

9.5. UI

For our UI design framework, we are incorporating both TailwindCSS and MUI to leverage the unique strengths of each. TailwindCSS has been chosen for its efficient UI construction and component management. Its performance is notably superior, outperforming traditional CSS by 48%, as highlighted by Austin in 2023. This significant improvement makes it an optimal choice for crafting responsive and visually appealing interfaces with less effort. (Austin, 2023)

MUI complements our design strategy by providing a robust set of React components. This integration is essential for achieving a sleek and attractive UI design, facilitating the development of complex UI features with simplicity and elegance.

By combining TailwindCSS's efficiency and MUI's comprehensive component library, we aim to create a user interface that is not only beautiful and user-friendly but also highly performant and scalable. This dual-framework approach allows us to harness the best of both worlds, ensuring a smooth and responsive user experience across our application.

9.6. Real time updating

For implementing real-time updates, we're planning to integrate WebSockets into our system. This technology enables instant communication between the user's browser and our server, ensuring that whenever a new request hits the database, the user on our website will receive an immediate notification via a popup.

Using WebSockets might be a bit more sophisticated than other methods for achieving real-time updates, but I've successfully utilized it in a previous project. This experience gives me confidence that incorporating it into our current web application will significantly enhance the user experience by keeping everyone informed with the latest updates as they happen.

Another approach might be to use React Querry. I've only heard about this method once and there is not a lot of documentation available, but this might also be a valid path to achieve the same result. Real Time updating.

9.7. Push alerts

To implement push alerts in our application, we're planning to focus on two main channels: email and LINE push notifications. There are others but these two seem to be the two most useful.

For sending email notifications, there are several tools and services we can utilize. These tools enable us to automate the process of sending out email alerts whenever specific actions occur within our application, such as new requests or updates.

For LINE notifications, we'll need to integrate with the LINE Messaging API. This requires setting up a LINE developer account, creating a LINE bot, and configuring it to send notifications. With this setup, we can send real-time alerts directly to a user's LINE app, informing them of new developments or updates relevant to their interests or actions on our platform.

By leveraging both email and LINE for push notifications, we aim to offer our users a flexible and efficient way to stay informed about important events and updates, enhancing their overall experience with our application.

9.8. Authentication

In evaluating our options for implementing authentication, we've narrowed down our choices to two primary candidates: Firebase and Auth0.

For our project, customization is a key requirement, especially the ability to incorporate our own fields for login and registration processes. After careful consideration, Firebase emerges as the more suitable choice due to its superior customization capabilities. It allows for the integration of custom user fields directly within the authentication flow, enabling us to tailor the login and registration experience to our specific needs. This flexibility extends to using Firebase's other services, such as Cloud Firestore or Realtime Database, to store and manage additional user data beyond the standard authentication parameters.

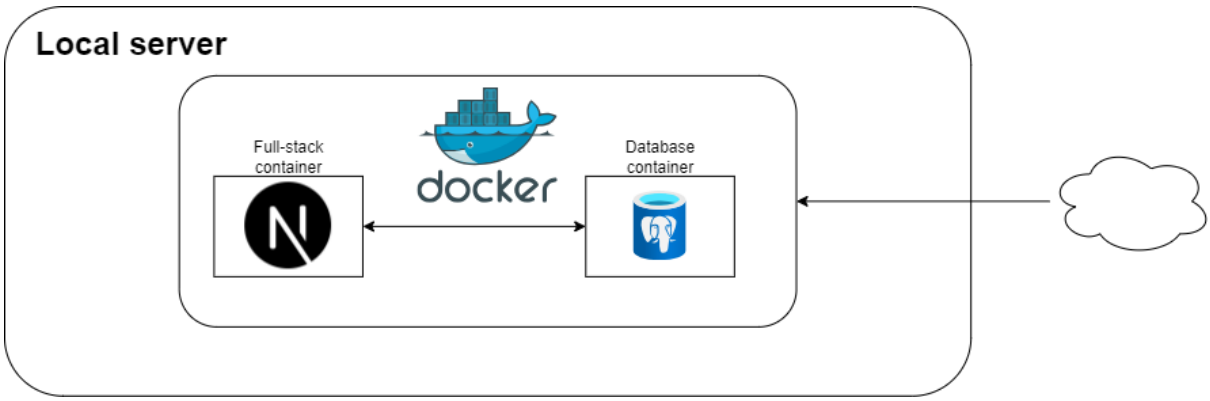
While Auth0 also offers a high degree of customization and a user-friendly interface for managing authentication flows, it seems that Firebase provides a more granular level of control that aligns better with our project's requirements. This includes the ability to create completely custom user interfaces for authentication, without the need for redirecting users to external pages, and the option to handle custom data more seamlessly within our application's infrastructure.

Moreover, Firebase's integration with other Google Cloud Platform services could offer additional advantages in terms of scalability, data processing, and analytics, which may prove beneficial as our project evolves.

In summary, although Auth0 is an exceptional choice with strong features for managing authentication flows, Firebase's extensive customization options and seamless integration with custom fields and additional user data make it the preferred choice for our project's specific needs. This decision is guided by our priority to deliver a tailored and integrated user experience throughout the authentication process.

9.9. Infrastructure

Here is what our infrastructure will look like:



The infrastructure for this application will be very simple because does it not require a big and complicated infrastructure.

The infrastructure for this application will look as the picture above suggests. On the local server there is a vSphere hypervisor installed, on this hypervisor I will create a Linux server. On this server I will install docker, and with this I will create 2 containers, 1 for the Next.js application itself and another for the PostgreSQL database. This way the application is easily scalable in the future if that is what the university wants.

9.9.1. Hosting

VMware vSphere®

vSphere is a software suite developed by VMware, offering virtualization and cloud computing solutions for businesses. It facilitates the creation, management, and optimization of virtualized infrastructure and cloud environments. Key components include the ESXi Hypervisor, which partitions physical servers into multiple virtual machines (VMs); vCenter Server for centralized management and monitoring; the vSphere Client for web-based interface access; vSphere High Availability (HA) for automated failover protection; vSphere Distributed Resource Scheduler (DRS) for workload balancing; and vSphere Storage, providing virtualized storage solutions. Overall, vSphere simplifies the management and optimization of IT resources, granting businesses increased flexibility, scalability, and efficiency in their operations.

9.9.2. Containers

Docker is a widely used platform for developing, shipping, and running applications within containers. Containers are lightweight, portable, and isolated environments that bundle an application with its dependencies. Docker simplifies the process of creating, deploying, and managing containers through its core components: Docker Engine, Dockerfile, Docker Image, Docker Container, Docker Hub, and Docker Compose. It allows developers to package applications into portable units called Docker images, which can be easily shared and run on any platform that supports Docker. Docker has significantly impacted software development by providing a standardized and efficient way to package, distribute, and run applications across different environments.

9.10. Security

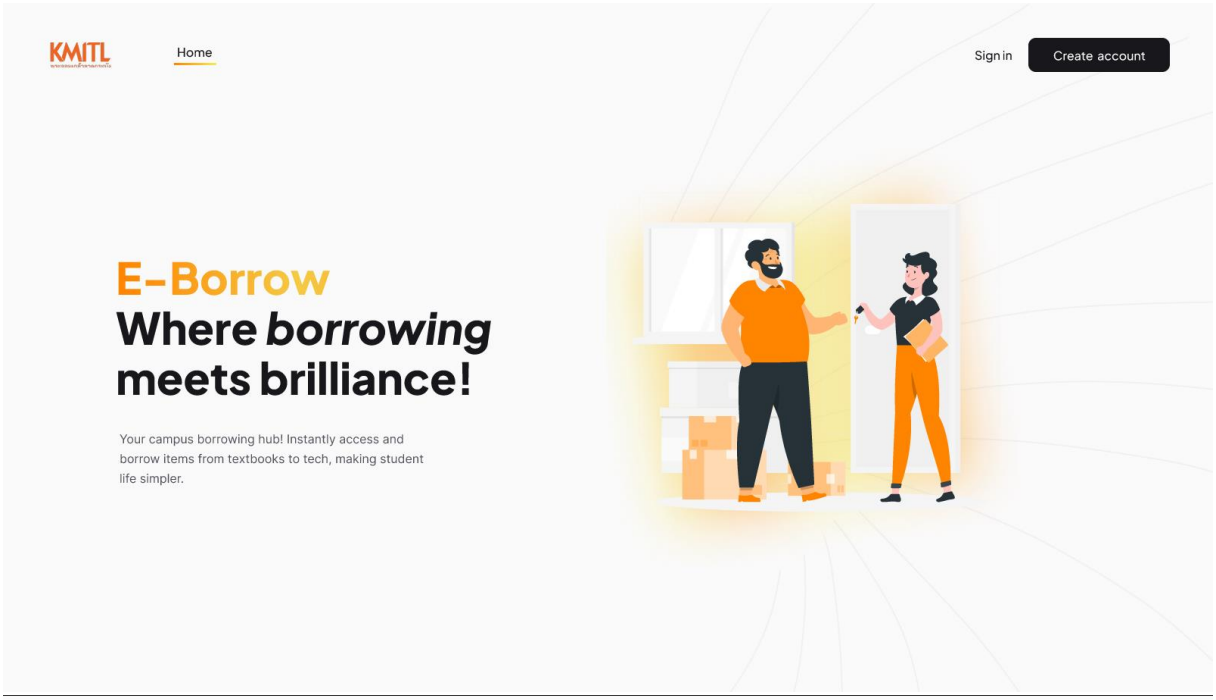
How the security measures will be implemented is mentioned in detail in the other file (Project_Plan_Security.dockx).

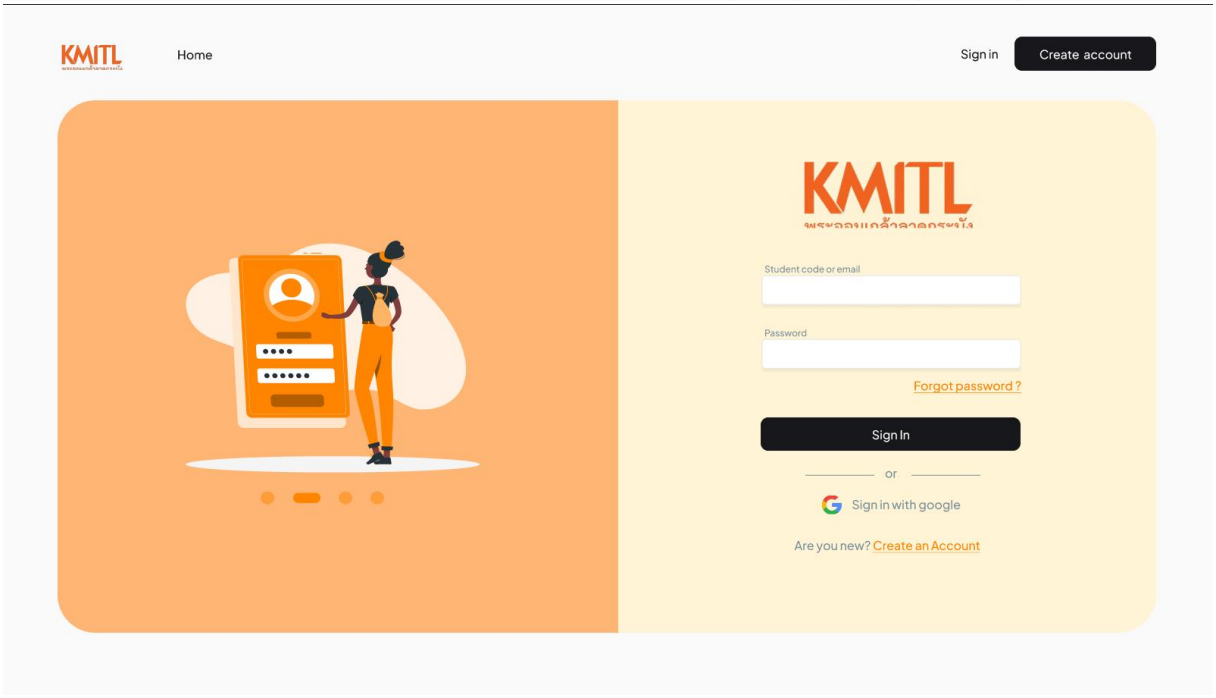
In simple terms the CCS student will play the role of an evil developer on the developer platform. He will clone the repository weekly and check for weaknesses on the platform while also trying to exploit the application. The information gotten from this will be used to secure the application.

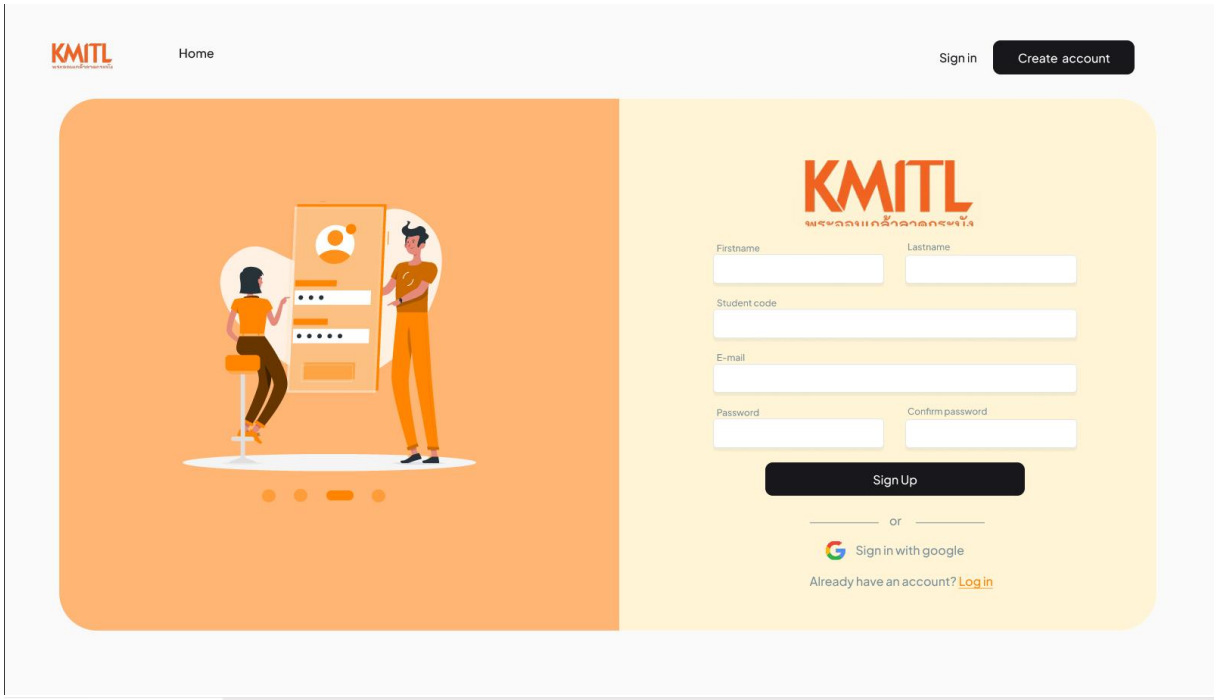
10. Storyboard

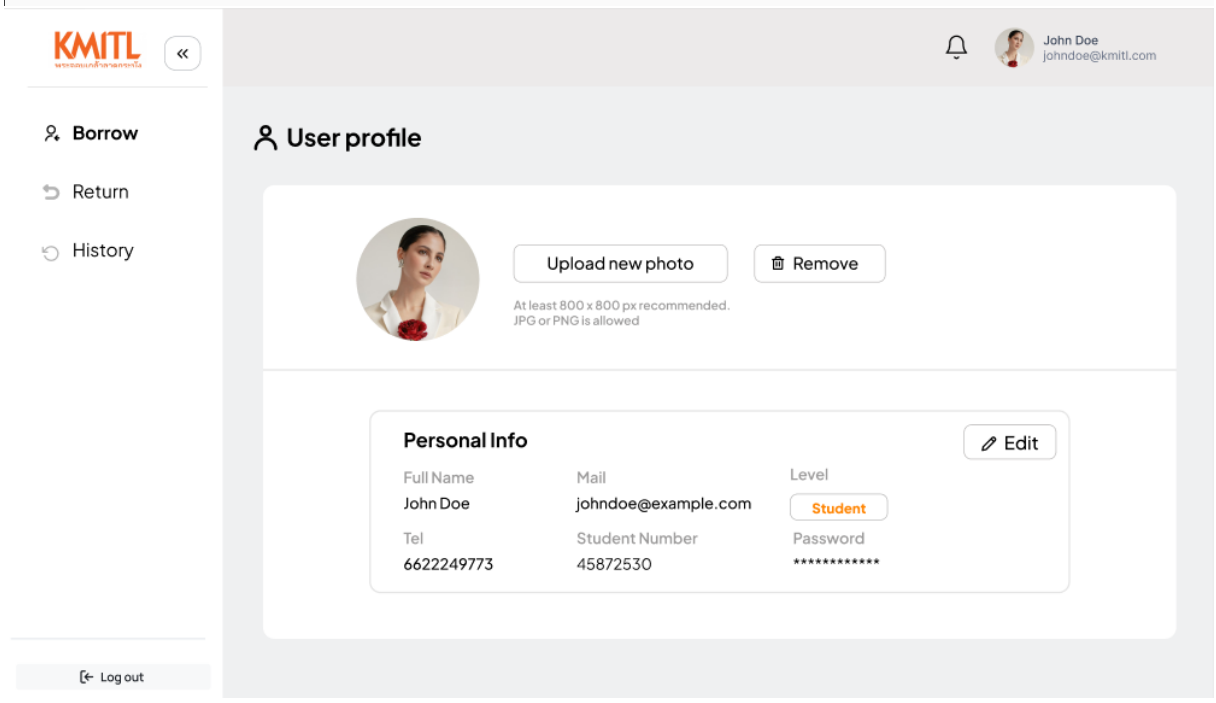
10.1. Desktop

10.1.1. General:

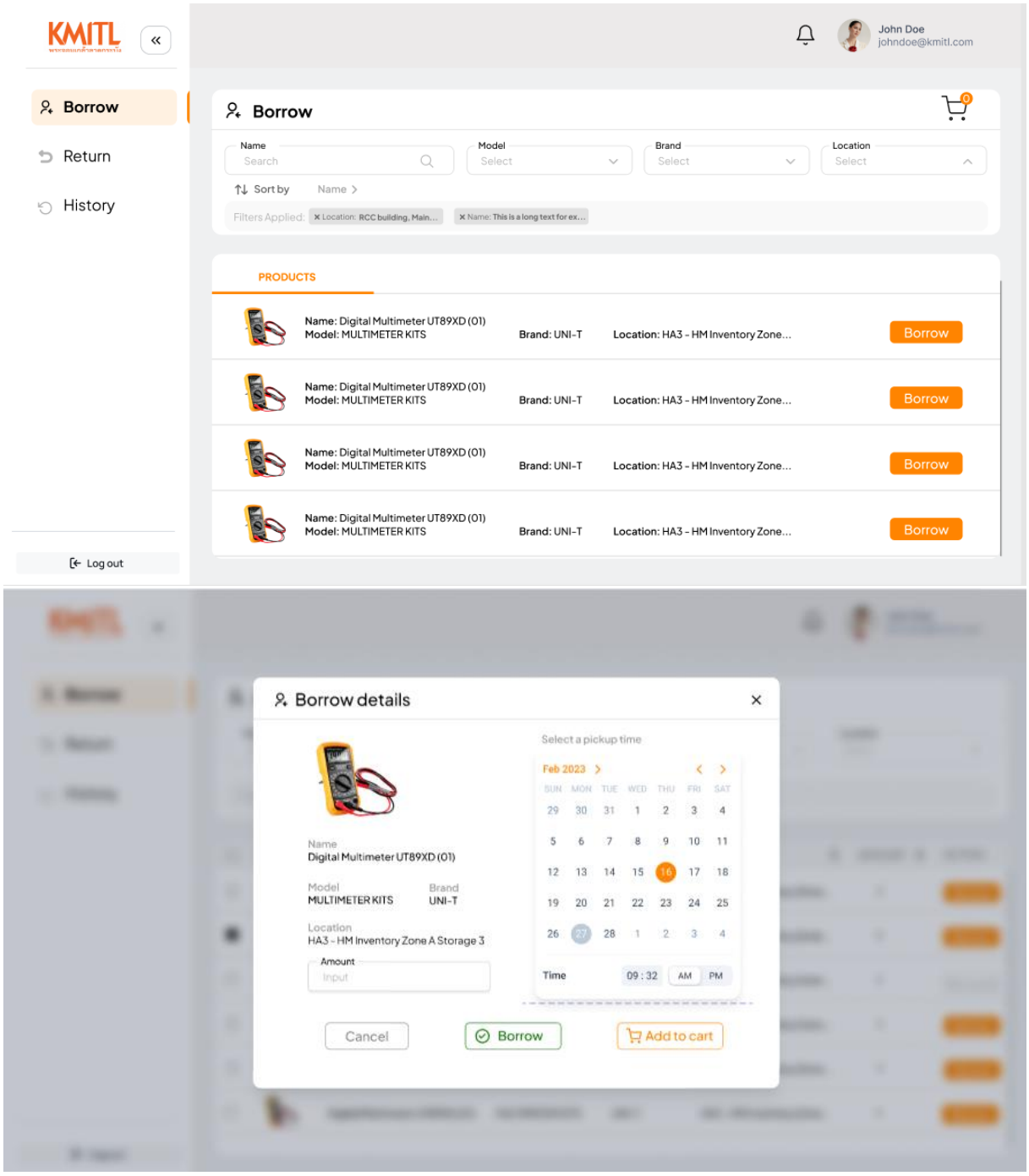


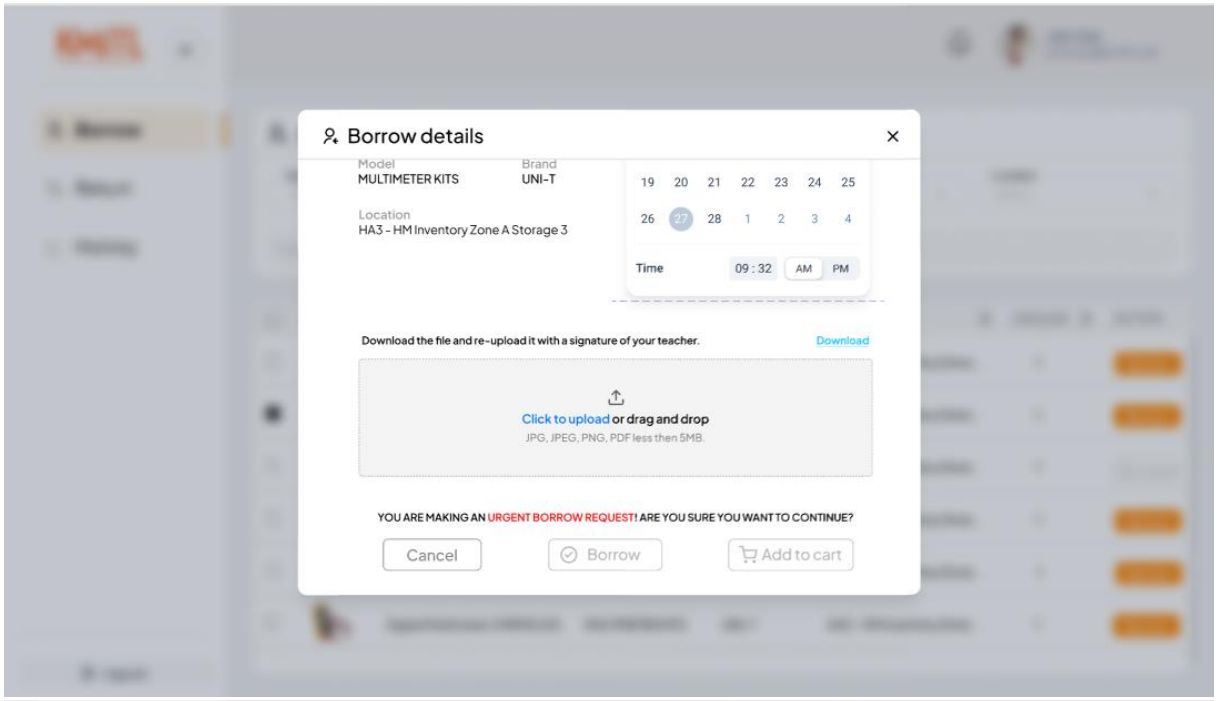


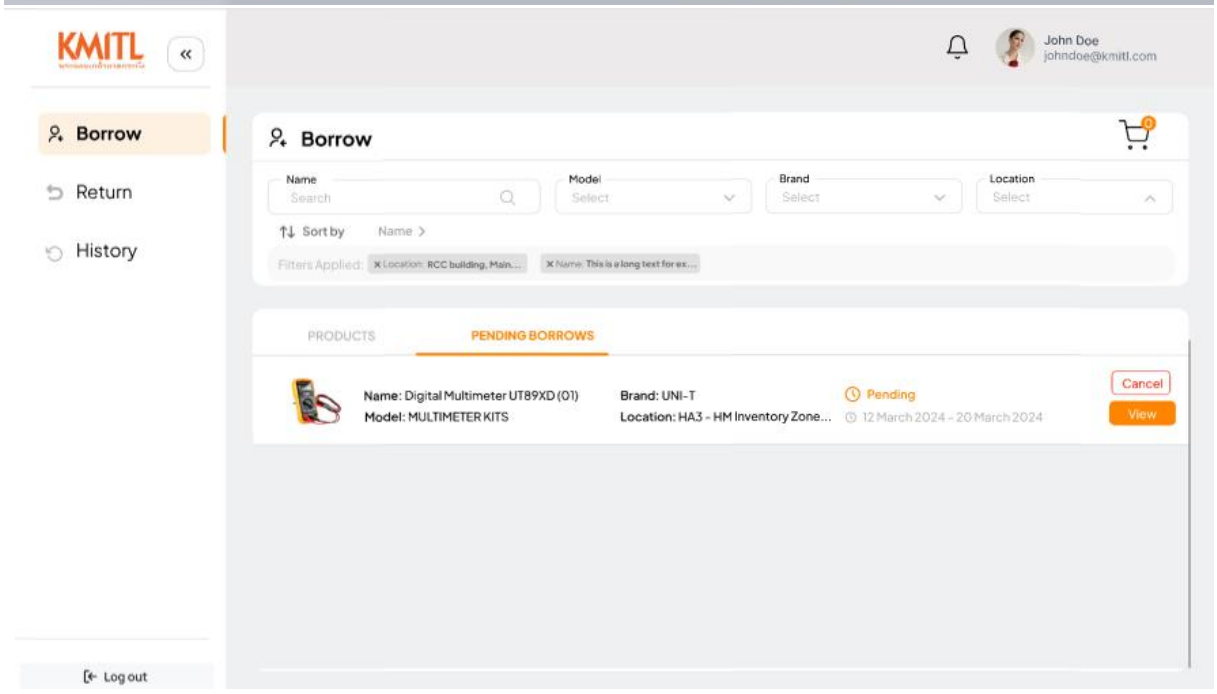

KMITL
Home
Sign in
Create account
KMITL
Firstname
Lastname
Student code
E-mail
Password
Confirm password
Sign Up
or
Sign in with google
Already have an account? Log in

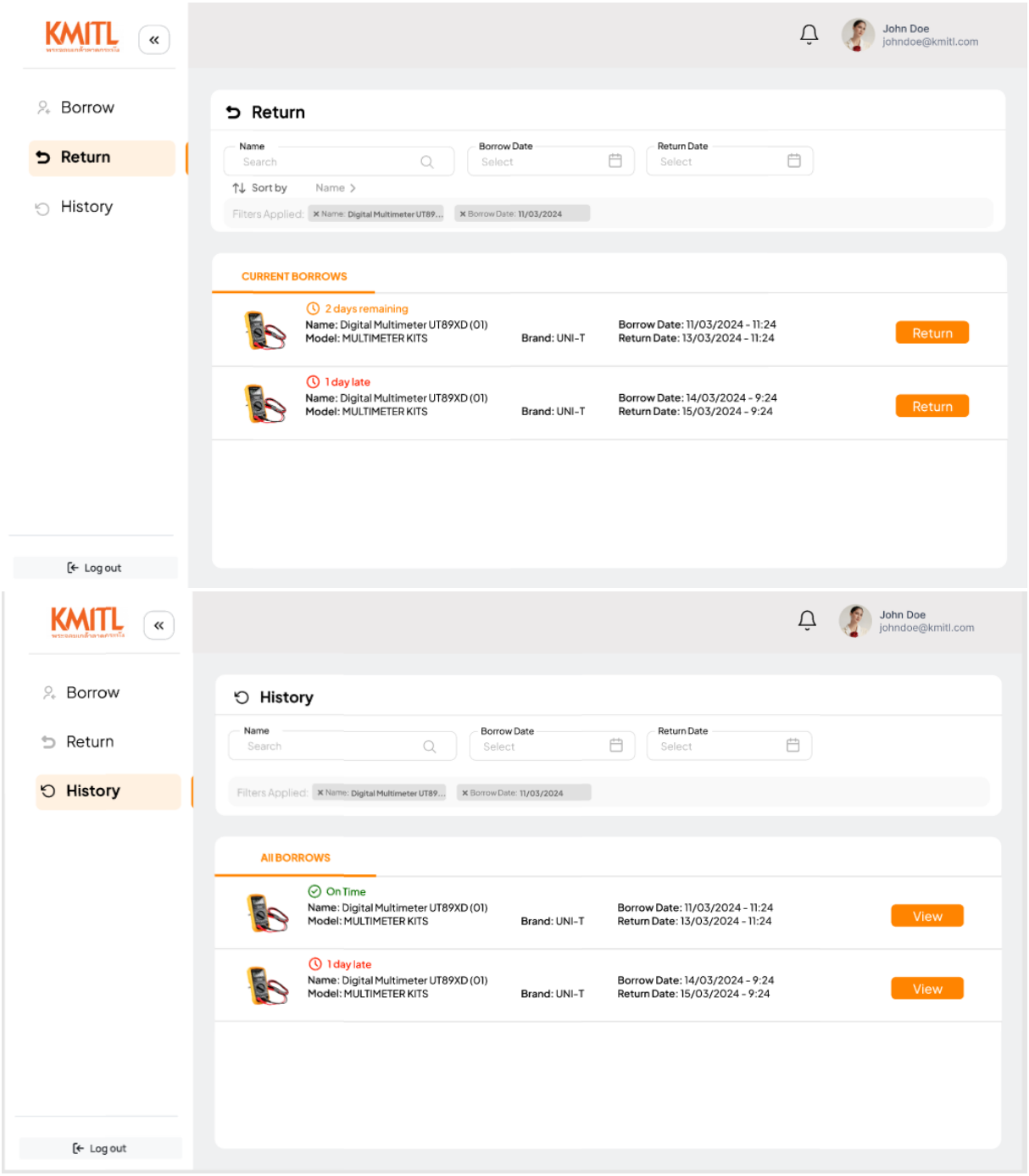

KMITL
John Doe
johndoe@kmitl.com
Borrow
Return
History
User profile
Upload new photo
Remove
At least 800 x 800 px recommended.
JPG or PNG is allowed
Personal Info
Edit
Full Name
John Doe
Mail
johndoe@example.com
Level
Student
Tel
6622249773
Student Number
45872530
Password
****
Log out

10.1.2. Student login:

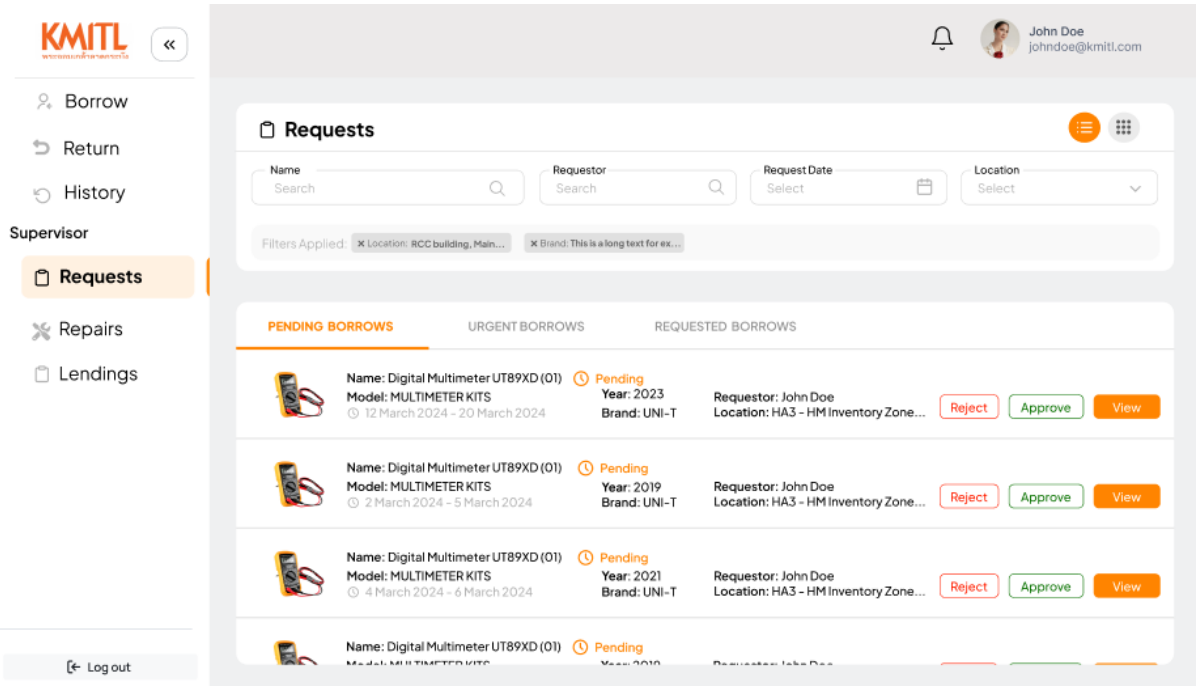


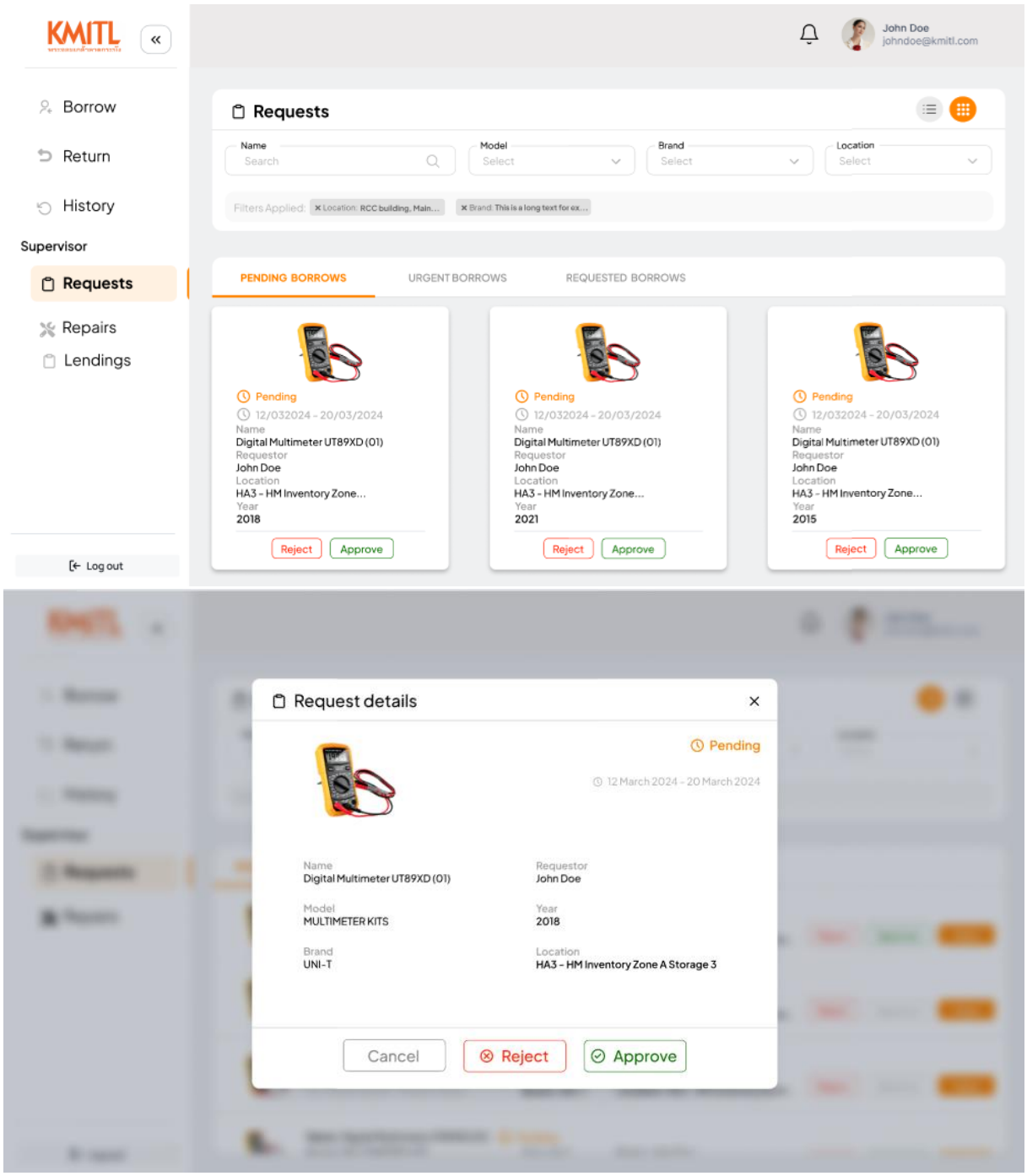

Borrow details
Model
MULTIMETER KITS
Brand
UNI-T
Location
HA3 - HM Inventory Zone A Storage 3
19 20 21 22 23 24 25
26 27 28 1 2 3 4
Time
09 : 32
AM
PM
Download the file and re-upload it with a signature of your teacher.
Download
Click to upload or drag and drop
JPG, JPEG, PNG, PDF less then 5MB.
YOU ARE MAKING AN URGENT BORROW REQUEST! ARE YOU SURE YOU WANT TO CONTINUE?
Cancel
Borrow
Add to cart

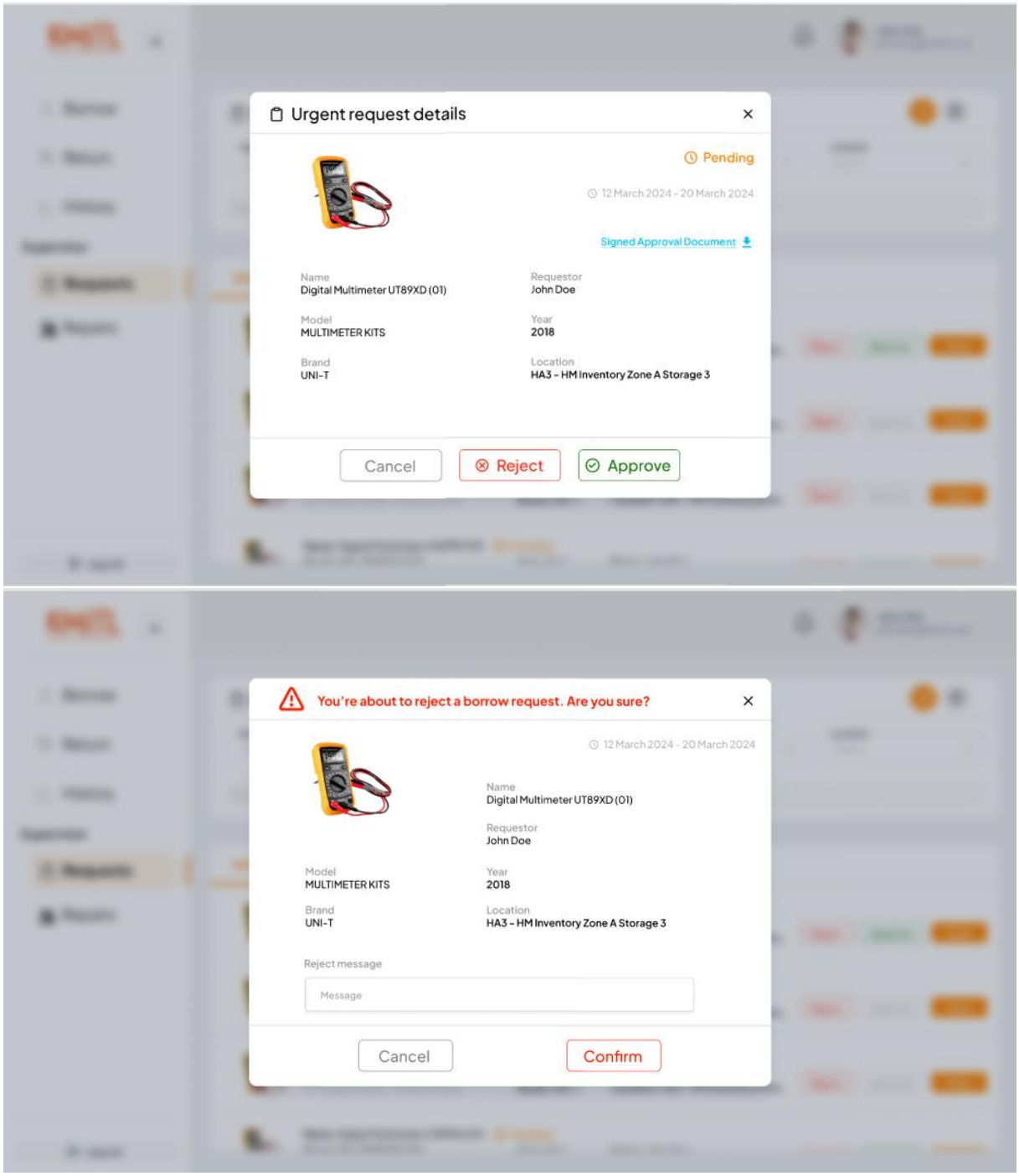

KMITL
John Doe
johndoe@kmitl.com
Borrow
Return
History
Log out
Borrow
Name
Search
Model
Select
Brand
Select
Location
Select
Sort by
Name
Filters Applied:
PRODUCTS
PENDING BORROWS
Name: Digital Multimeter UT89XD (01)
Model: MULTIMETER KITS
Brand: UNI-T
Location: HA3 - HM Inventory Zone...
Pending
12 March 2024 - 20 March 2024
Cancel
View

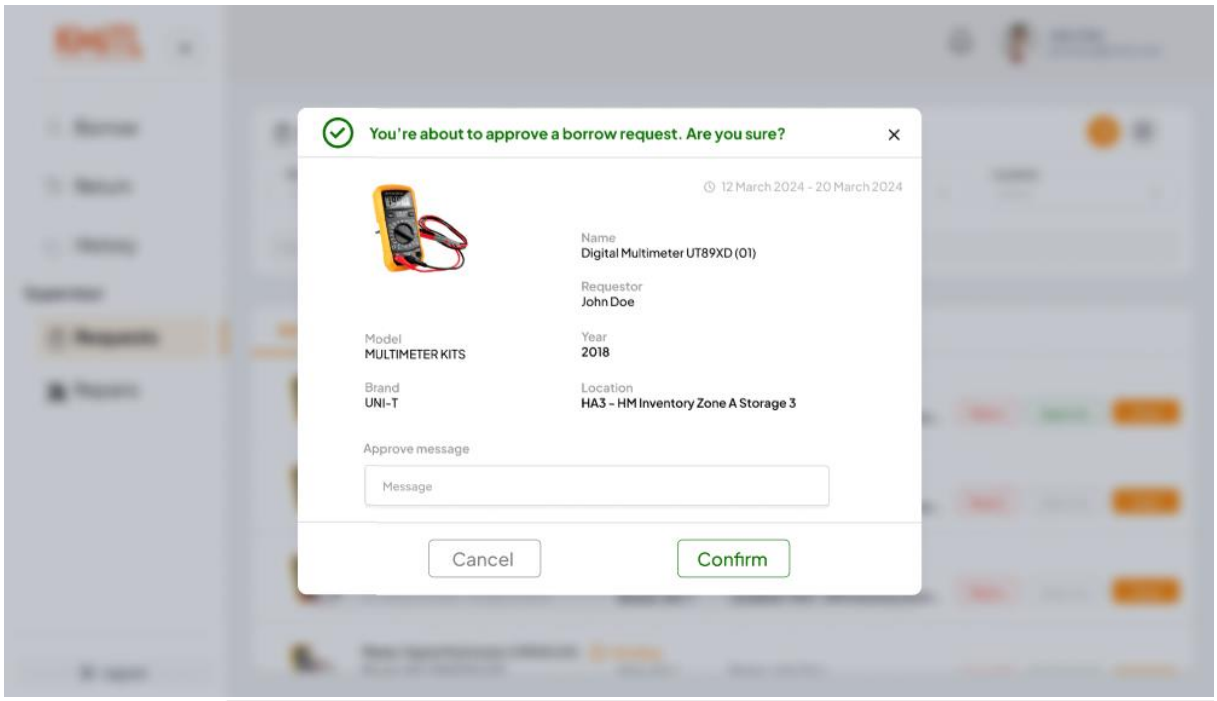

KMITL
John Doe
johndoe@kmitl.com
Borrow
Return
History
Log out
Return
Name
Search
Borrow Date
Select
Return Date
Select
Sort by
Name
Filters Applied:
Name: Digital Multimeter UT89...
Borrow Date: 11/03/2024
CURRENT BORROWS
2 days remaining
Name: Digital Multimeter UT89XD (01)
Model: MULTIMETER KITS
Brand: UNI-T
Borrow Date: 11/03/2024 - 11:24
Return Date: 13/03/2024 - 11:24
Return
1 day late
Name: Digital Multimeter UT89XD (01)
Model: MULTIMETER KITS
Brand: UNI-T
Borrow Date: 14/03/2024 - 9:24
Return Date: 15/03/2024 - 9:24
Return
KMITL
John Doe
johndoe@kmitl.com
Borrow
Return
History
Log out
History
Name
Search
Borrow Date
Select
Return Date
Select
Filters Applied:
Name: Digital Multimeter UT89...
Borrow Date: 11/03/2024
All BORROWS
On Time
Name: Digital Multimeter UT89XD (01)
Model: MULTIMETER KITS
Brand: UNI-T
Borrow Date: 11/03/2024 - 11:24
Return Date: 13/03/2024 - 11:24
View
1 day late
Name: Digital Multimeter UT89XD (01)
Model: MULTIMETER KITS
Brand: UNI-T
Borrow Date: 14/03/2024 - 9:24
Return Date: 15/03/2024 - 9:24
View

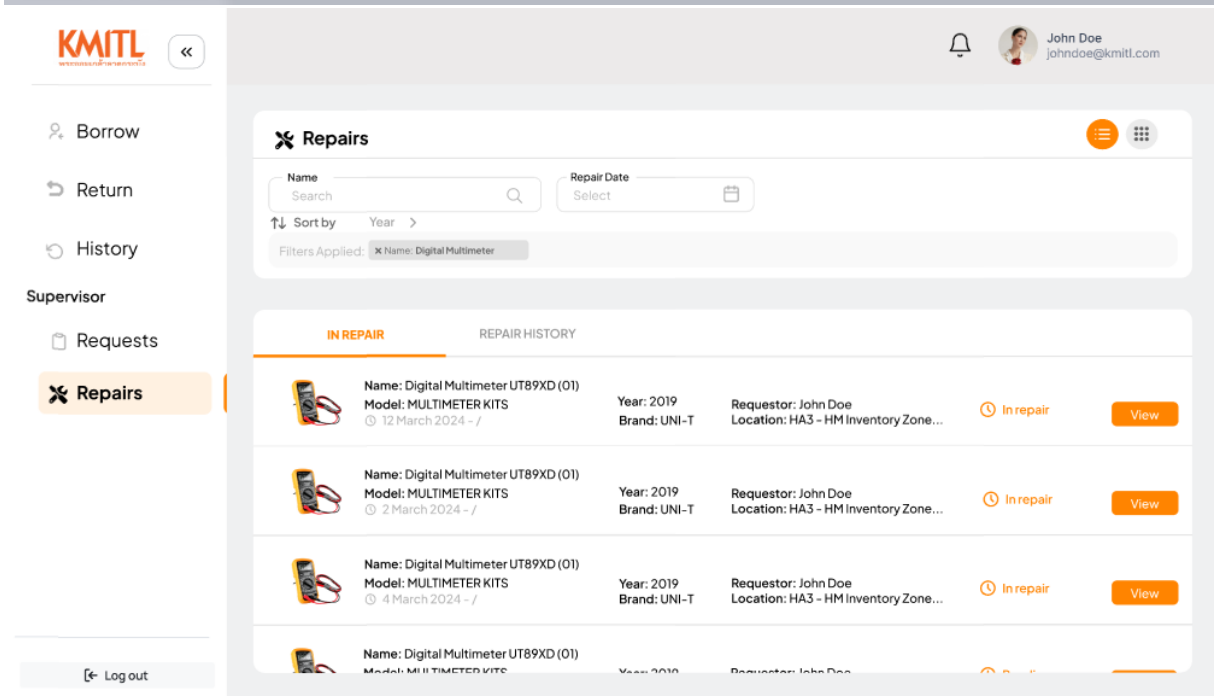
10.1.3. Supervisor login:

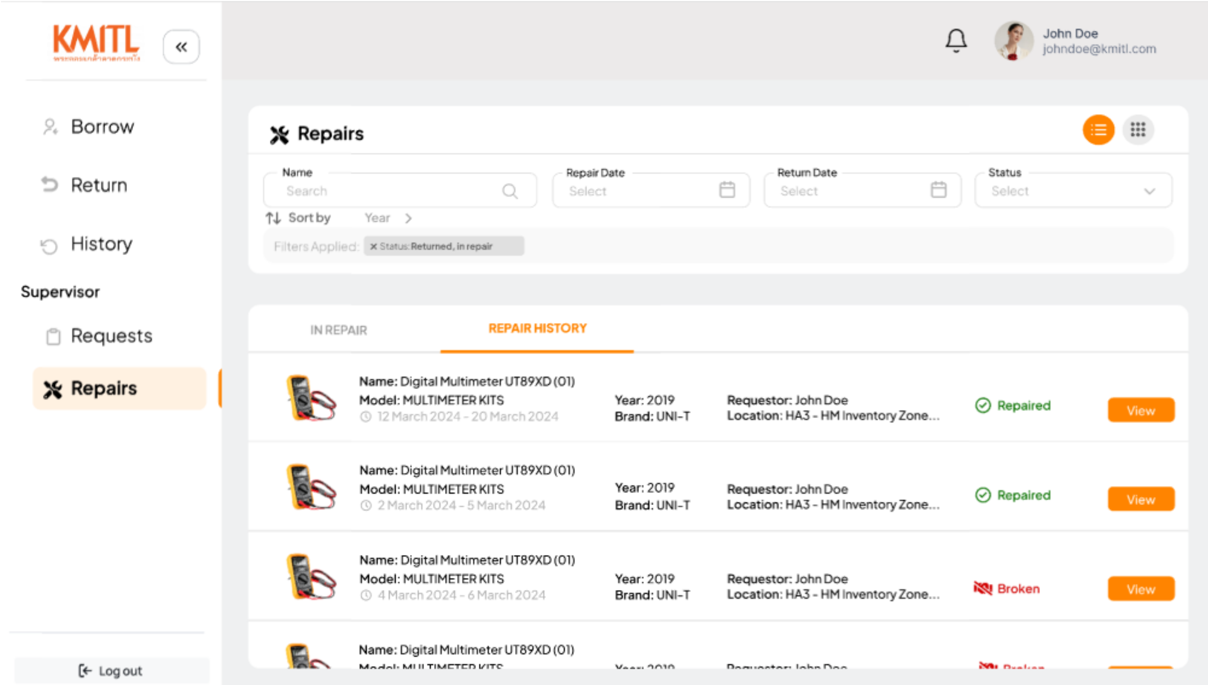


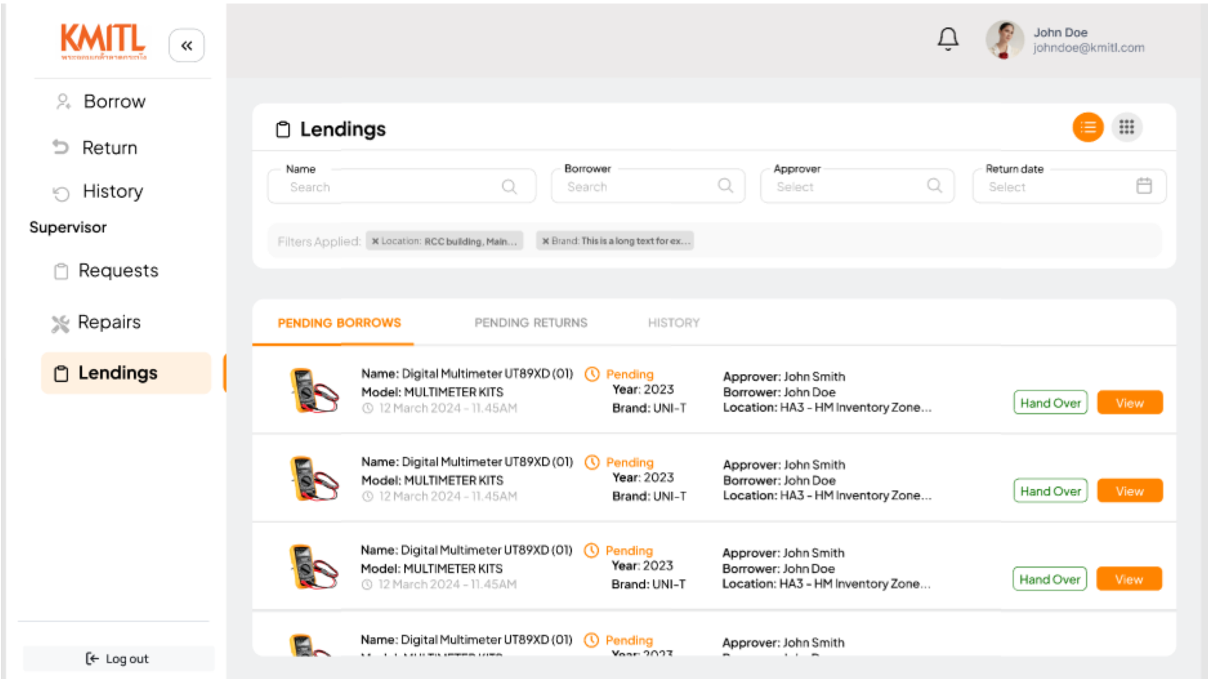

KMITL
John Doe
johndoe@kmitl.com
Borrow
Return
History
Supervisor
Requests
Repairs
Lendings
Log out
Requests
Name
Search
Model
Select
Brand
Select
Location
Select
Filters Applied:
Location: RCC building, Main...
Brand: This is a long text for ex...
PENDING BORROWS
URGENT BORROWS
REQUESTED BORROWS
Pending
12/032024 - 20/03/2024
Name
Digital Multimeter UT89XD (01)
Requestor
John Doe
Location
HA3 - HM Inventory Zone...
Year
2018
Reject
Approve
Pending
12/032024 - 20/03/2024
Name
Digital Multimeter UT89XD (01)
Requestor
John Doe
Location
HA3 - HM Inventory Zone...
Year
2021
Reject
Approve
Pending
12/032024 - 20/03/2024
Name
Digital Multimeter UT89XD (01)
Requestor
John Doe
Location
HA3 - HM Inventory Zone...
Year
2015
Reject
Approve
Request details
Pending
12 March 2024 - 20 March 2024
Name
Digital Multimeter UT89XD (01)
Requestor
John Doe
Model
MULTIMETER KITS
Year
2018
Brand
UNI-T
Location
HA3 - HM Inventory Zone A Storage 3
Cancel
Reject
Approve

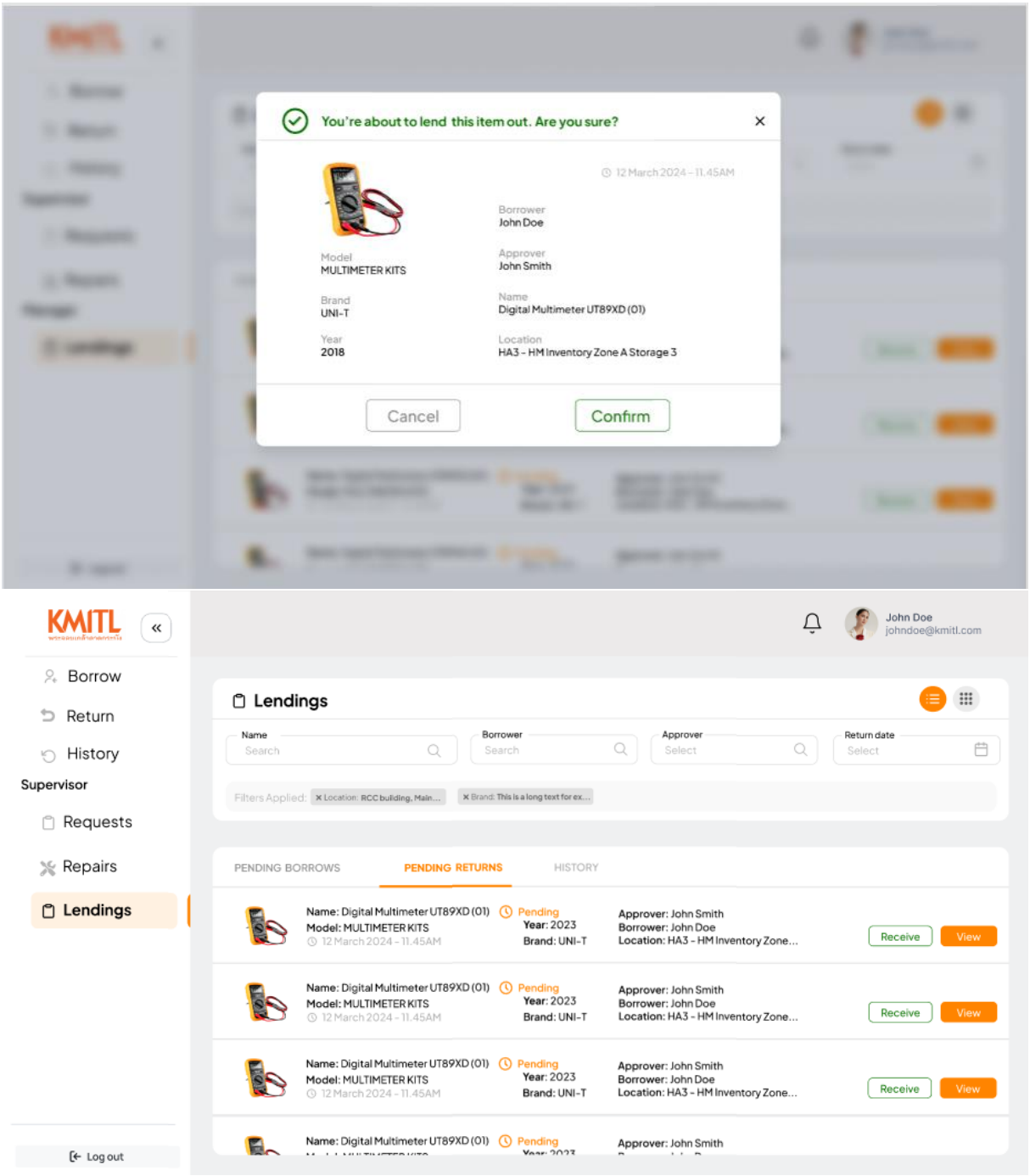

Urgent request details
Pending
12 March 2024 - 20 March 2024
Signed Approval Document
Name
Digital Multimeter UT89XD (01)
Requestor
John Doe
Model
MULTIMETER KITS
Year
2018
Brand
UNI-T
Location
HA3 - HM Inventory Zone A Storage 3
Cancel
Reject
Approve
You're about to reject a borrow request. Are you sure?
12 March 2024 - 20 March 2024
Name
Digital Multimeter UT89XD (01)
Requestor
John Doe
Model
MULTIMETER KITS
Year
2018
Brand
UNI-T
Location
HA3 - HM Inventory Zone A Storage 3
Reject message
Message
Cancel
Confirm

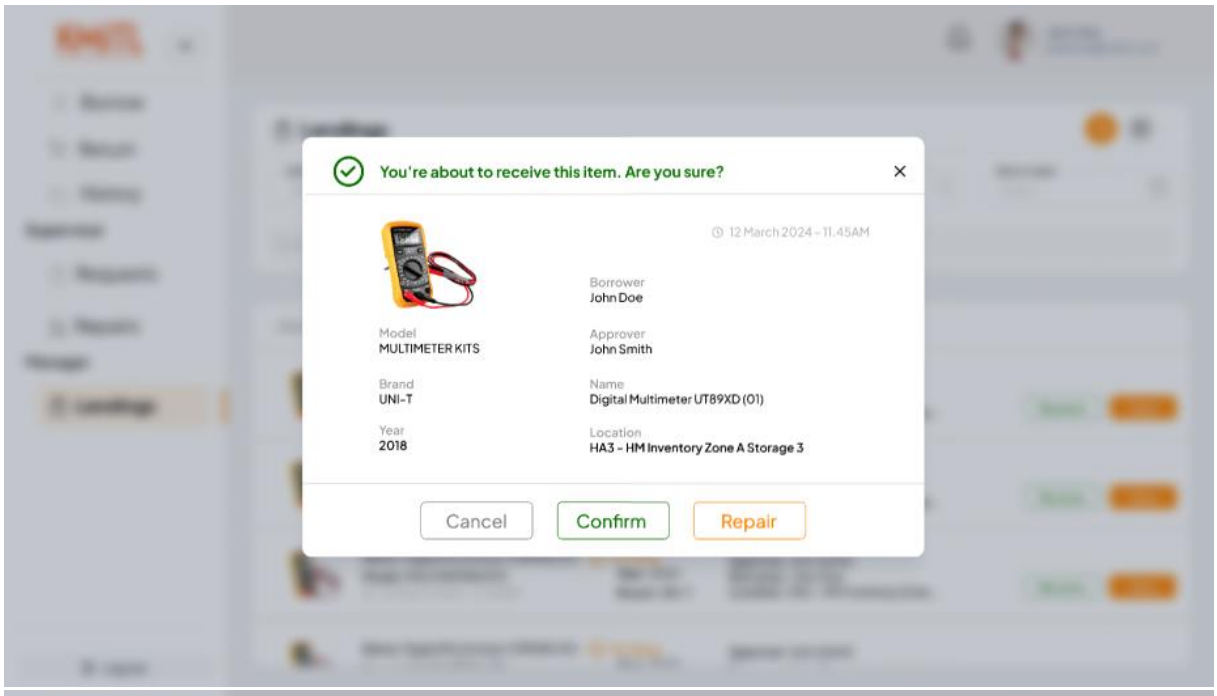

You're about to approve a borrow request. Are you sure?
12 March 2024 - 20 March 2024
Name
Digital Multimeter UT89XD (01)
Requestor
John Doe
Model
MULTIMETER KITS
Year
2018
Brand
UNI-T
Location
HA3 - HM Inventory Zone A Storage 3
Approve message
Message
Cancel
Confirm

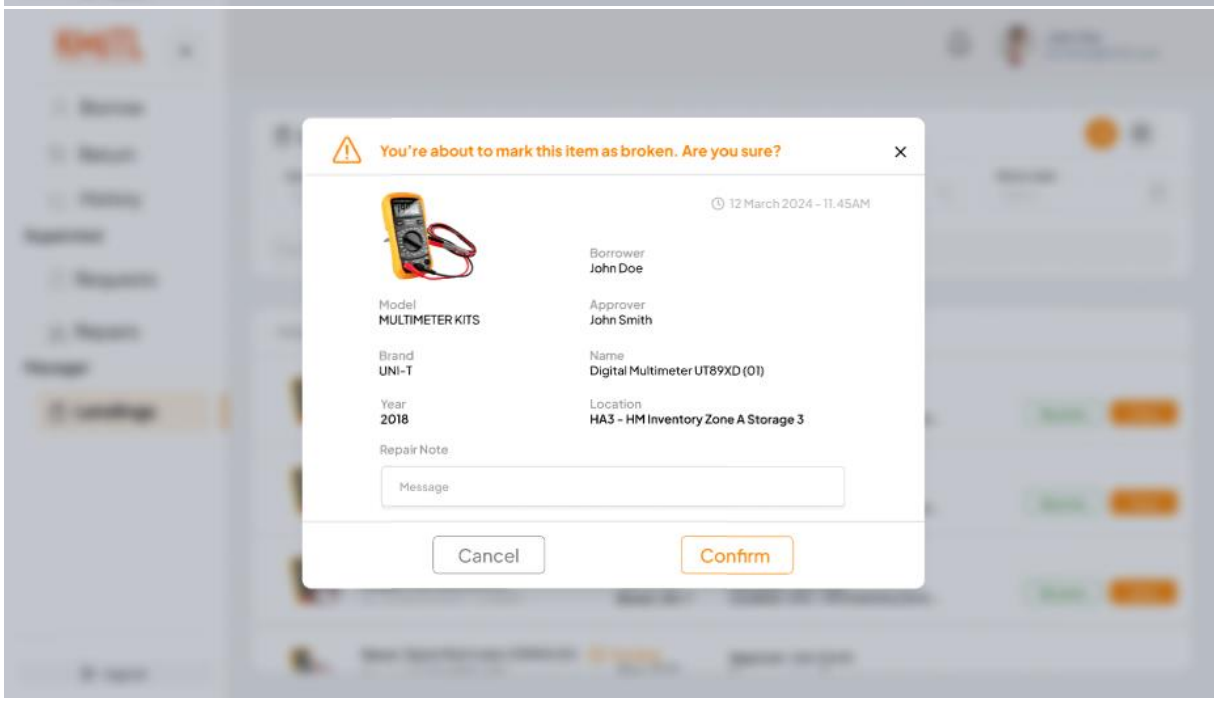

KMITL
John Doe
johndoe@kmitl.com
Borrow
Return
History
Supervisor
Requests
Repairs
Log out
Repairs
Name
Search
Repair Date
Select
Sort by
Year
Filters Applied:
Name: Digital Multimeter
IN REPAIR
REPAIR HISTORY
Name: Digital Multimeter UT89XD (01)
Model: MULTIMETER KITS
12 March 2024 - /
Year: 2019
Brand: UNI-T
Requestor: John Doe
Location: HA3 - HM Inventory Zone...
In repair
View
Name: Digital Multimeter UT89XD (01)
Model: MULTIMETER KITS
2 March 2024 - /
Year: 2019
Brand: UNI-T
Requestor: John Doe
Location: HA3 - HM Inventory Zone...
In repair
View
Name: Digital Multimeter UT89XD (01)
Model: MULTIMETER KITS
4 March 2024 - /
Year: 2019
Brand: UNI-T
Requestor: John Doe
Location: HA3 - HM Inventory Zone...
In repair
View
Name: Digital Multimeter UT89XD (01)

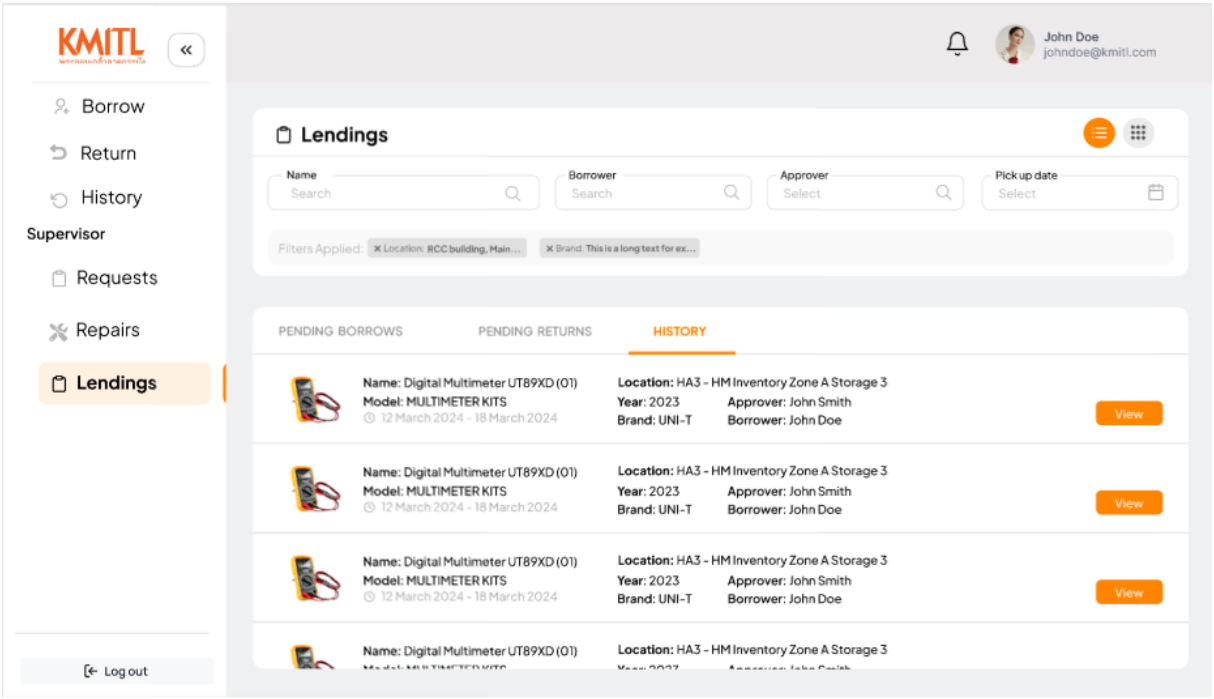

KMITL
Borrow
Return
History
Supervisor
Requests
Repairs
Log out
John Doe
johndoe@kmitl.com
Repairs
Name
Search
Repair Date
Select
Return Date
Select
Status
Select
Sort by
Year
Filters Applied:
Status: Returned, In repair
IN REPAIR
REPAIR HISTORY
Name: Digital Multimeter UT89XD (01)
Model: MULTIMETER KITS
12 March 2024 - 20 March 2024
Year: 2019
Brand: UNI-T
Requestor: John Doe
Location: HA3 - HM Inventory Zone...
Repaired
View
Name: Digital Multimeter UT89XD (01)
Model: MULTIMETER KITS
2 March 2024 - 5 March 2024
Year: 2019
Brand: UNI-T
Requestor: John Doe
Location: HA3 - HM Inventory Zone...
Repaired
View
Name: Digital Multimeter UT89XD (01)
Model: MULTIMETER KITS
4 March 2024 - 6 March 2024
Year: 2019
Brand: UNI-T
Requestor: John Doe
Location: HA3 - HM Inventory Zone...
Broken
View
Name: Digital Multimeter UT89XD (01)


KMITL
Borrow
Return
History
Supervisor
Requests
Repairs
Lendings
Log out
John Doe
johndoe@kmitl.com
Lendings
Name
Search
Borrower
Search
Approver
Select
Return date
Select
Filters Applied:
Location: RCC building, Main...
Brand: This is a long text for ex...
PENDING BORROWS
PENDING RETURNS
HISTORY
Name: Digital Multimeter UT89XD (01)
Pending
Model: MULTIMETER KITS
12 March 2024 - 11.45AM
Year: 2023
Brand: UNI-T
Approver: John Smith
Borrower: John Doe
Location: HA3 - HM Inventory Zone...
Hand Over
View
Name: Digital Multimeter UT89XD (01)
Pending
Model: MULTIMETER KITS
12 March 2024 - 11.45AM
Year: 2023
Brand: UNI-T
Approver: John Smith
Borrower: John Doe
Location: HA3 - HM Inventory Zone...
Hand Over
View
Name: Digital Multimeter UT89XD (01)
Pending
Model: MULTIMETER KITS
12 March 2024 - 11.45AM
Year: 2023
Brand: UNI-T
Approver: John Smith
Borrower: John Doe
Location: HA3 - HM Inventory Zone...
Hand Over
View
Name: Digital Multimeter UT89XD (01)
Pending
Approver: John Smith

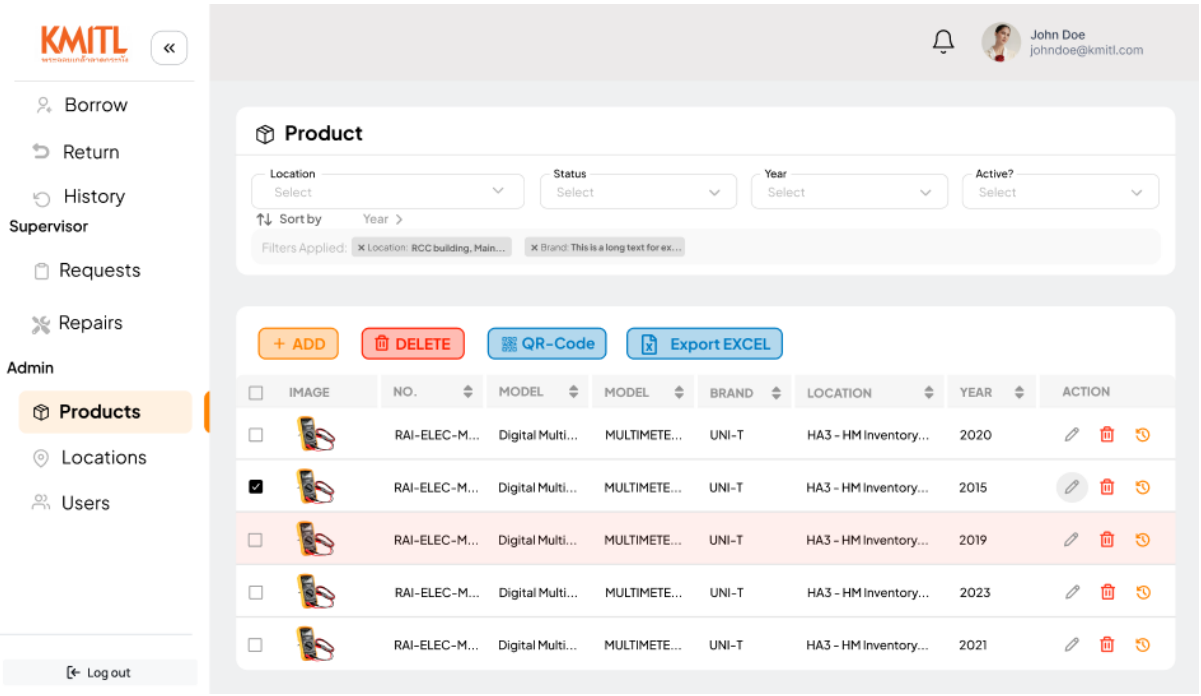

You're about to lend this item out. Are you sure?
12 March 2024 - 11.45AM
Borrower
John Doe
Approver
John Smith
Model
MULTIMETER KITS
Name
Digital Multimeter UT89XD (01)
Brand
UNI-T
Year
2018
Location
HA3 - HM Inventory Zone A Storage 3
Cancel
Confirm
KMITL
John Doe
johndoe@kmitl.com
Borrow
Return
History
Supervisor
Requests
Repairs
Lendings
Log out
Lendings
Name
Search
Borrower
Search
Approver
Select
Return date
Select
Filters Applied:
PENDING BORROWS
PENDING RETURNS
HISTORY
Name: Digital Multimeter UT89XD (01)
Model: MULTIMETER KITS
12 March 2024 - 11.45AM
Pending
Year: 2023
Brand: UNI-T
Approver: John Smith
Borrower: John Doe
Location: HA3 - HM Inventory Zone...
Receive
View

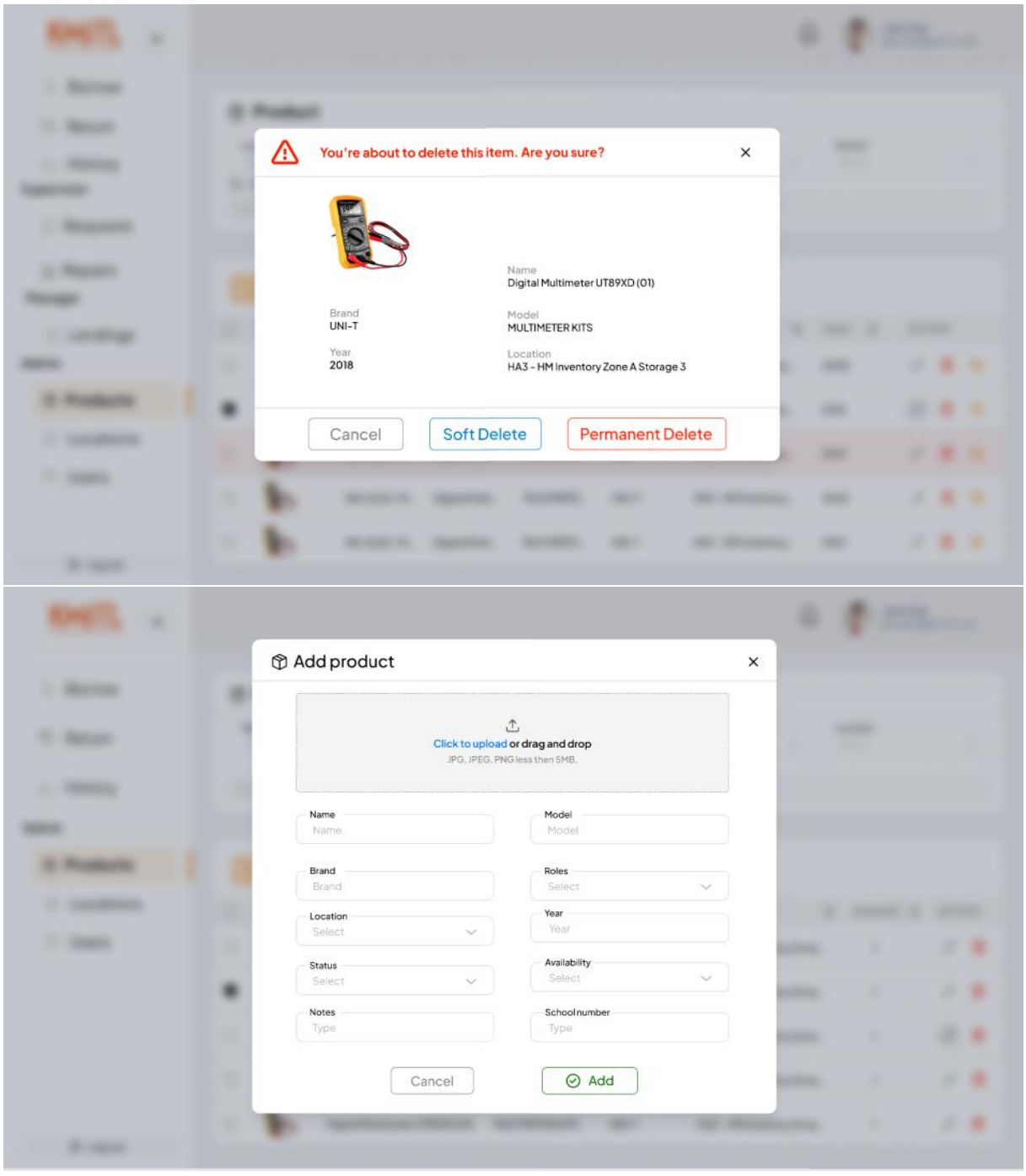

You're about to receive this item. Are you sure?
12 March 2024 - 11.45AM
Borrower
John Doe
Model
MULTIMETER KITS
Approver
John Smith
Brand
UNI-T
Name
Digital Multimeter UT89XD (01)
Year
2018
Location
HA3 - HM Inventory Zone A Storage 3
Cancel
Confirm
Repair

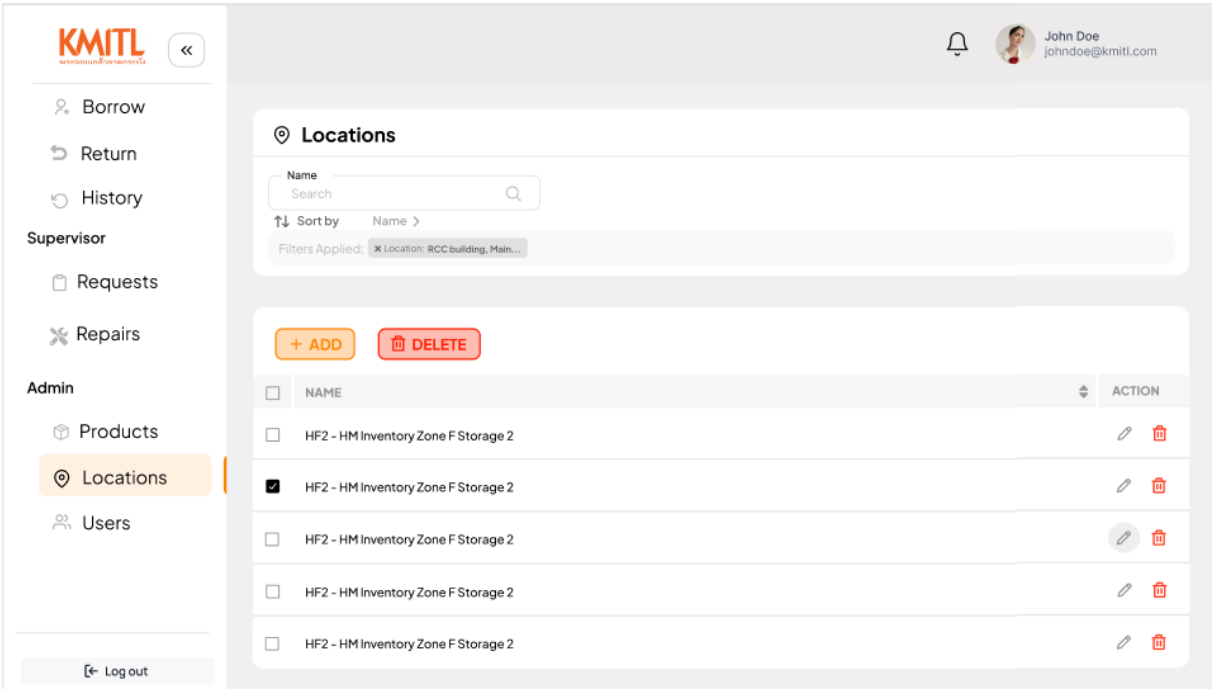

You're about to mark this item as broken. Are you sure?
12 March 2024 - 11.45AM
Borrower
John Doe
Model
MULTIMETER KITS
Approver
John Smith
Brand
UNI-T
Name
Digital Multimeter UT89XD (01)
Year
2018
Location
HA3 - HM Inventory Zone A Storage 3
Repair Note
Message
Cancel
Confirm

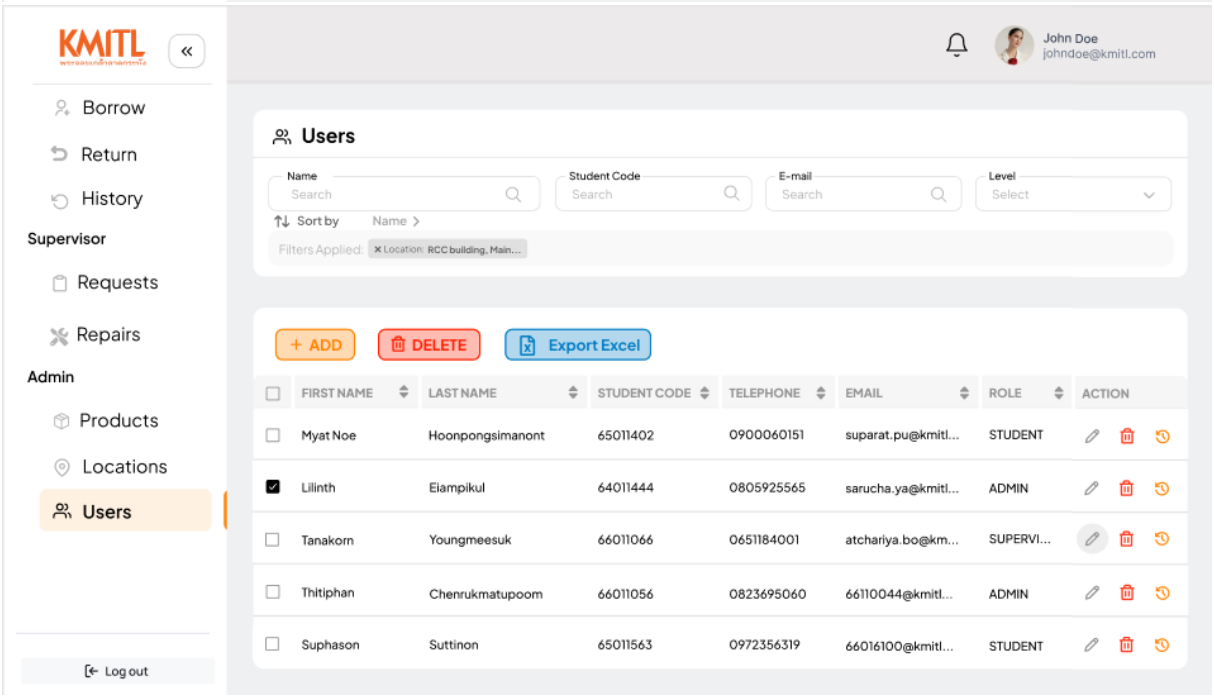


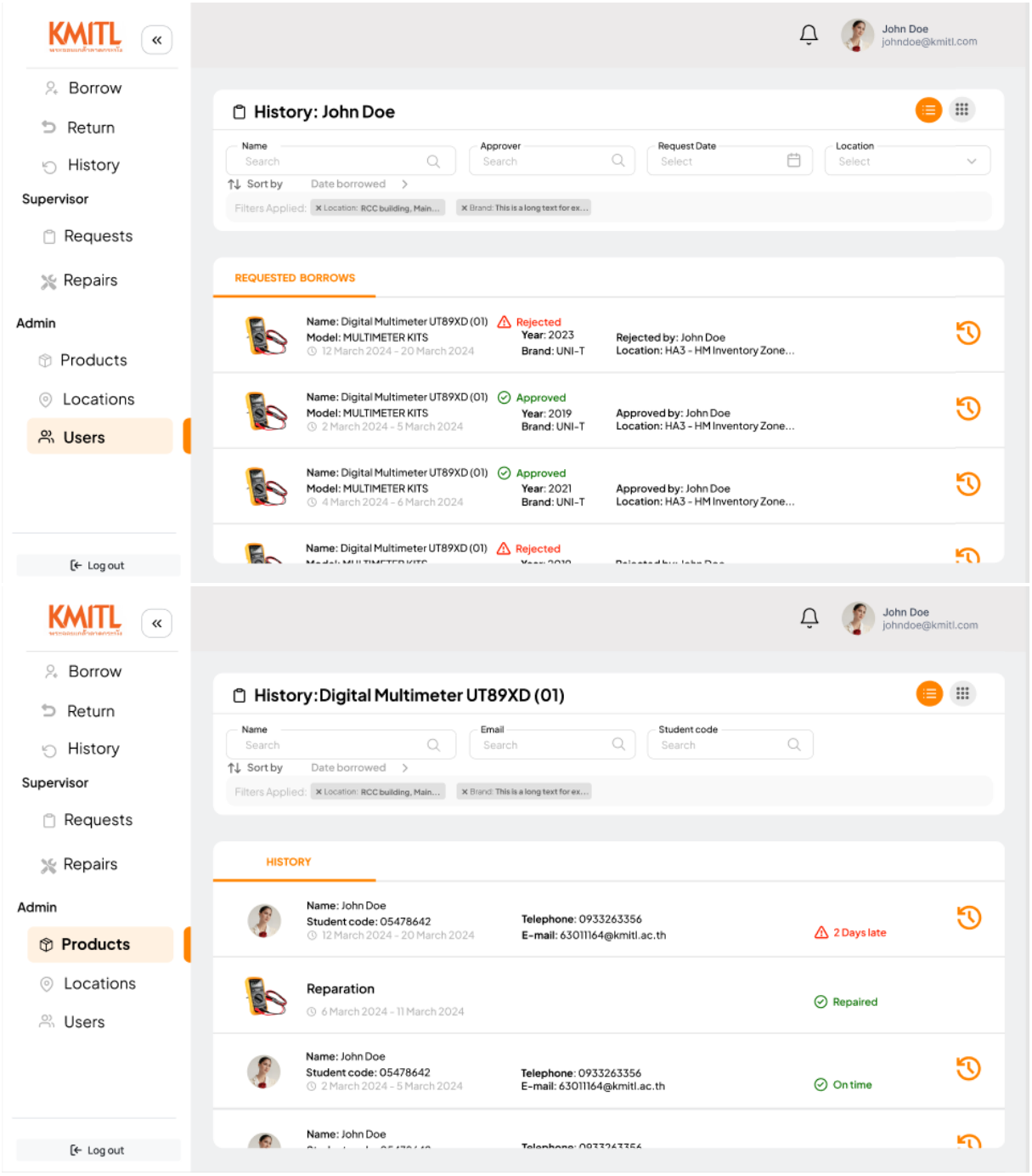
10.1.4. Admin login:




You're about to delete this item. Are you sure?
Name
Digital Multimeter UT89XD (01)
Brand
UNI-T
Model
MULTIMETER KITS
Year
2018
Location
HA3 - HM Inventory Zone A Storage 3
Cancel
Soft Delete
Permanent Delete
Add product
Click to upload or drag and drop
JPG, JPEG, PNG less then 5MB.
Name
Name
Model
Model
Brand
Brand
Roles
Select
Location
Select
Year
Year
Status
Select
Availability
Select
Notes
Type
School number
Type
Cancel
Add

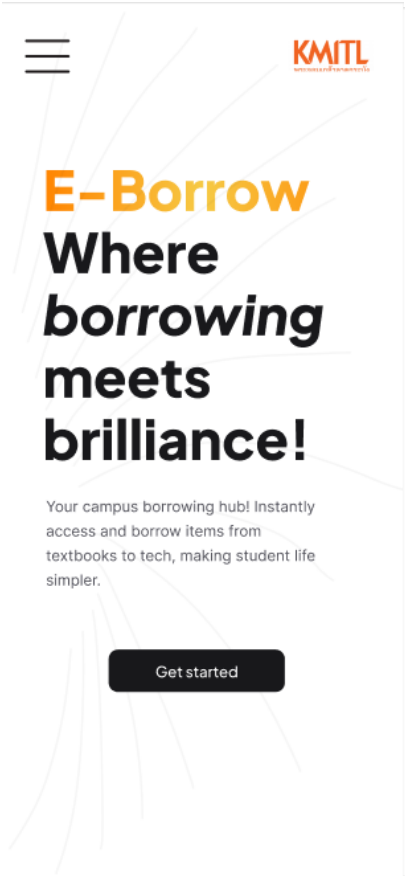

KMITL
Borrow
Return
History
Supervisor
Requests
Repairs
Admin
Products
Locations
Users
Log out
John Doe
johndoe@kmitl.com
Locations
Name
Search
Sort by
Name
Filters Applied:
Location: RCC building, Main...
+ ADD
DELETE
NAME
ACTION
HF2 - HM Inventory Zone F Storage 2
HF2 - HM Inventory Zone F Storage 2
HF2 - HM Inventory Zone F Storage 2
HF2 - HM Inventory Zone F Storage 2
HF2 - HM Inventory Zone F Storage 2

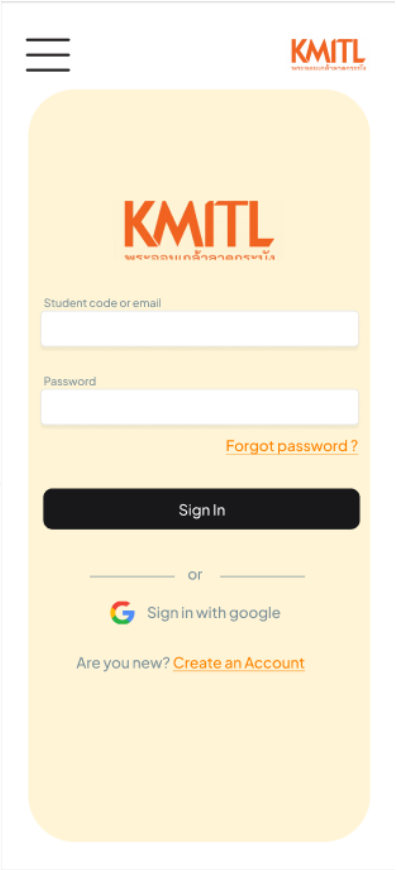

KMITL
Borrow
Return
History
Supervisor
Requests
Repairs
Admin
Products
Locations
Users
Log out
John Doe
johndoe@kmitl.com
Users
Name
Search
Student Code
Search
E-mail
Search
Level
Select
Sort by
Name
Filters Applied:
Location: RCC building, Main...
+ ADD
DELETE
Export Excel
FIRST NAME
LAST NAME
STUDENT CODE
TELEPHONE
EMAIL
ROLE
ACTION
Myat Noe
Hoonpongsimanont
65011402
0900060151
suparat.pu@kmitl...
STUDENT
Lilinth
Eiampikul
64011444
0805925565
sarucha.ya@kmitl...
ADMIN
Tanakorn
Youngmeesuk
66011066
0651184001
atchariya.bo@km...
SUPERVI...
Thitiphan
Chenrukmatupoom
66011056
0823695060
66110044@kmitl...
ADMIN
Suphason
Suttinon
65011563
0972356319
66016100@kmitl...
STUDENT

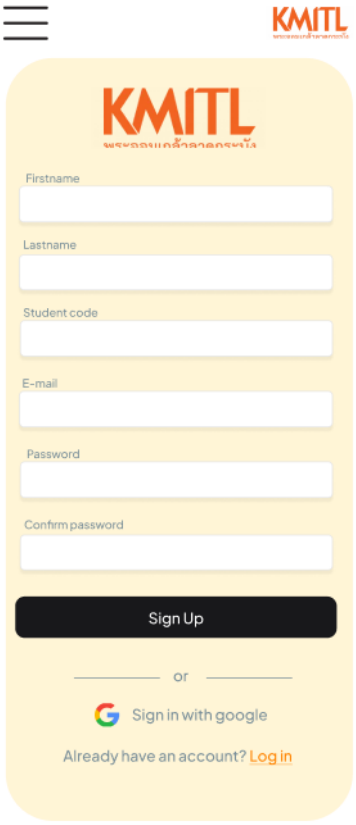

KMITL
Borrow
Return
History
Supervisor
Requests
Repairs
Admin
Products
Locations
Users
Log out
John Doe
johndoe@kmitl.com
History: John Doe
Name
Search
Approver
Search
Request Date
Select
Location
Select
Sort by
Date borrowed
Filters Applied:
Location: RCC building, Main...
Brand: This is a long text for ex...
REQUESTED BORROWS
Name: Digital Multimeter UT89XD (01)
Rejected
Model: MULTIMETER KITS
Year: 2023
Brand: UNI-T
12 March 2024 - 20 March 2024
Rejected by: John Doe
Location: HA3 - HM Inventory Zone...
Name: Digital Multimeter UT89XD (01)
Approved
Model: MULTIMETER KITS
Year: 2019
Brand: UNI-T
2 March 2024 - 5 March 2024
Approved by: John Doe
Location: HA3 - HM Inventory Zone...
Name: Digital Multimeter UT89XD (01)
Approved
Model: MULTIMETER KITS
Year: 2021
Brand: UNI-T
4 March 2024 - 6 March 2024
Approved by: John Doe
Location: HA3 - HM Inventory Zone...
Name: Digital Multimeter UT89XD (01)
Rejected
KMITL
Borrow
Return
History
Supervisor
Requests
Repairs
Admin
Products
Locations
Users
Log out
John Doe
johndoe@kmitl.com
History:Digital Multimeter UT89XD (01)
Name
Search
Email
Search
Student code
Search
Sort by
Date borrowed
Filters Applied:
Location: RCC building, Main...
Brand: This is a long text for ex...
HISTORY
Name: John Doe
Student code: 05478642
12 March 2024 - 20 March 2024
Telephone: 0933263356
E-mail: 63011164@kmitl.ac.th
2 Days late
Reparation
6 March 2024 - 11 March 2024
Repaired
Name: John Doe
Student code: 05478642
2 March 2024 - 5 March 2024
Telephone: 0933263356
E-mail: 63011164@kmitl.ac.th
On time
Name: John Doe

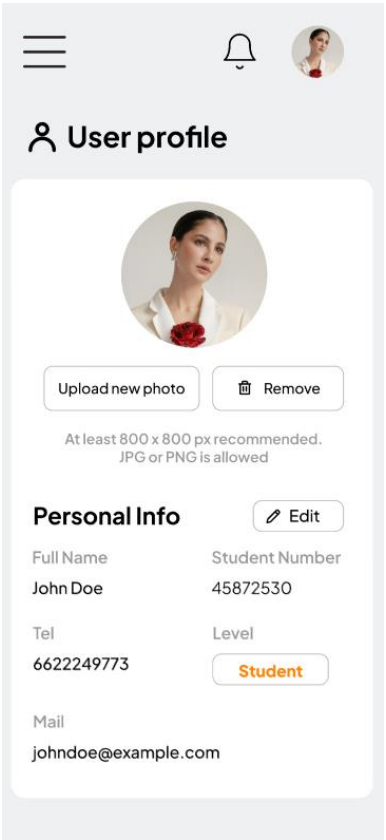
10.2. Mobile

10.2.1. General:

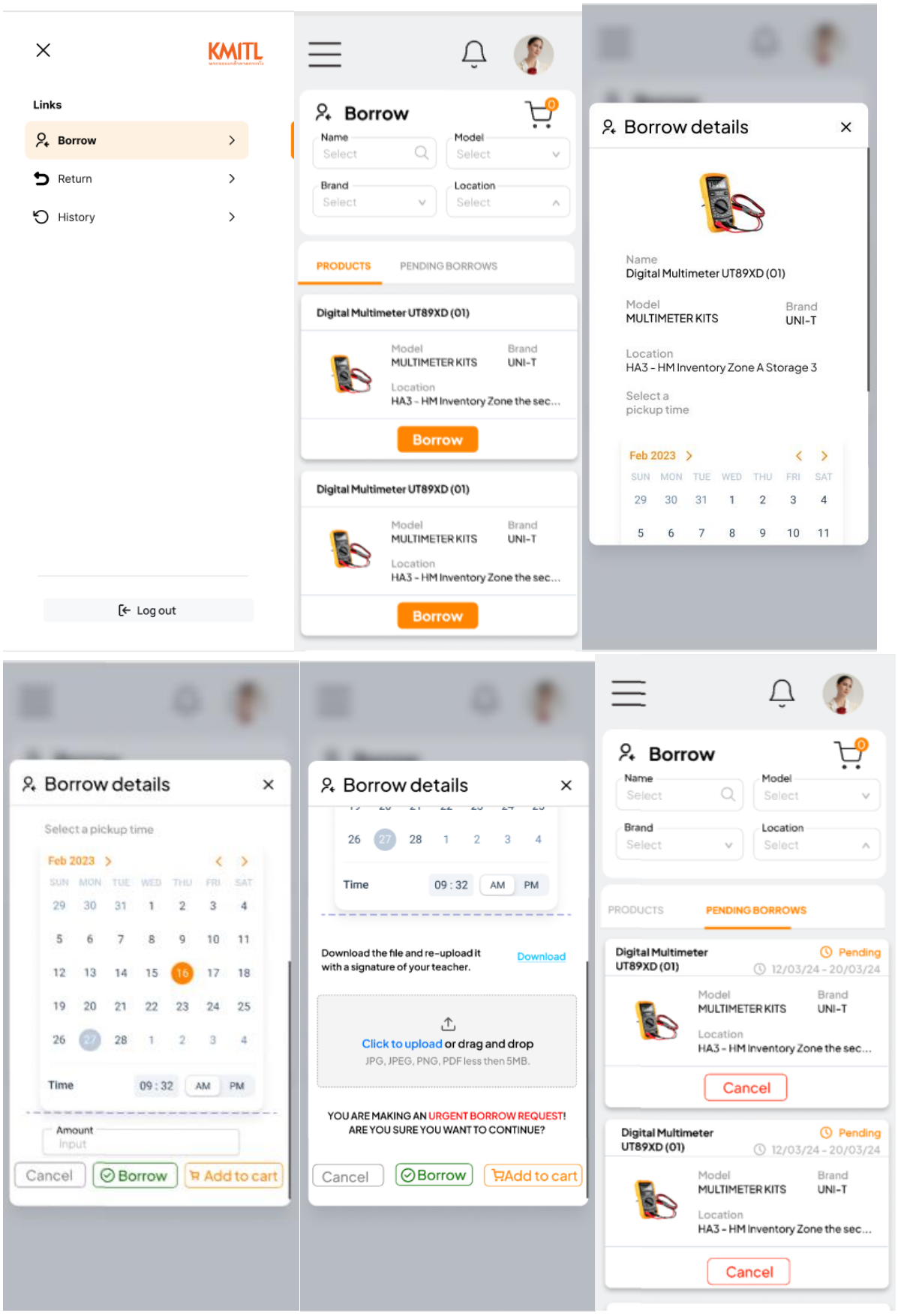


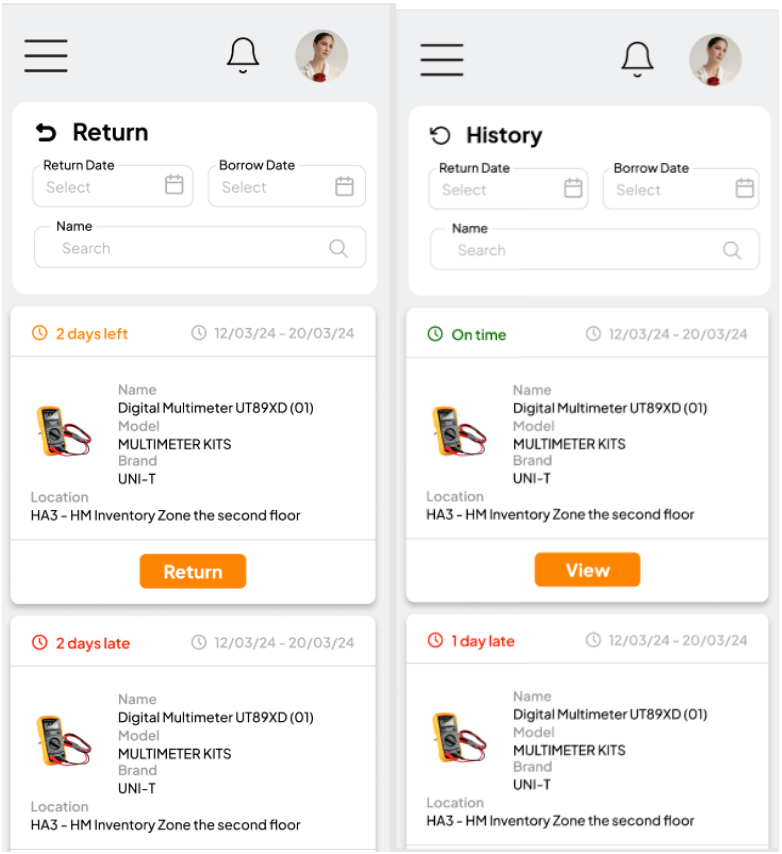




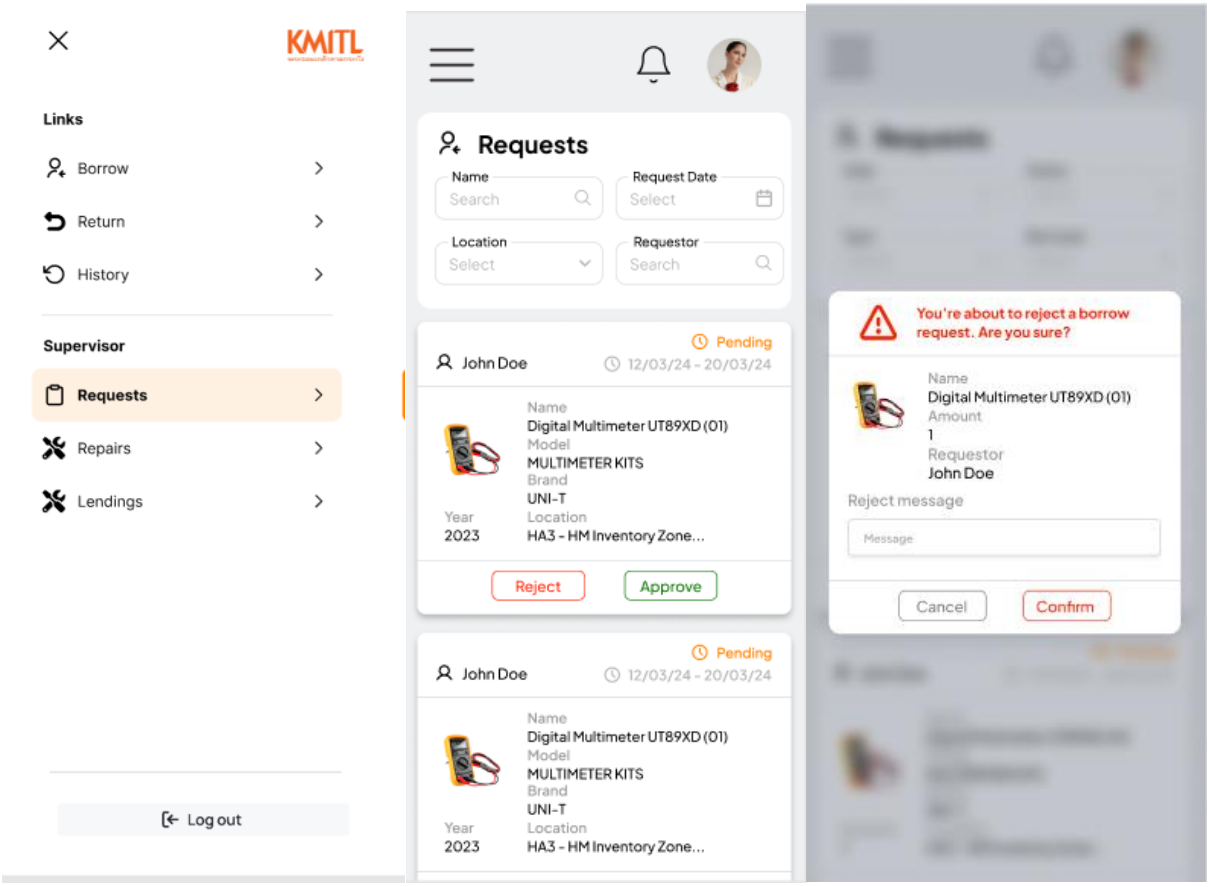


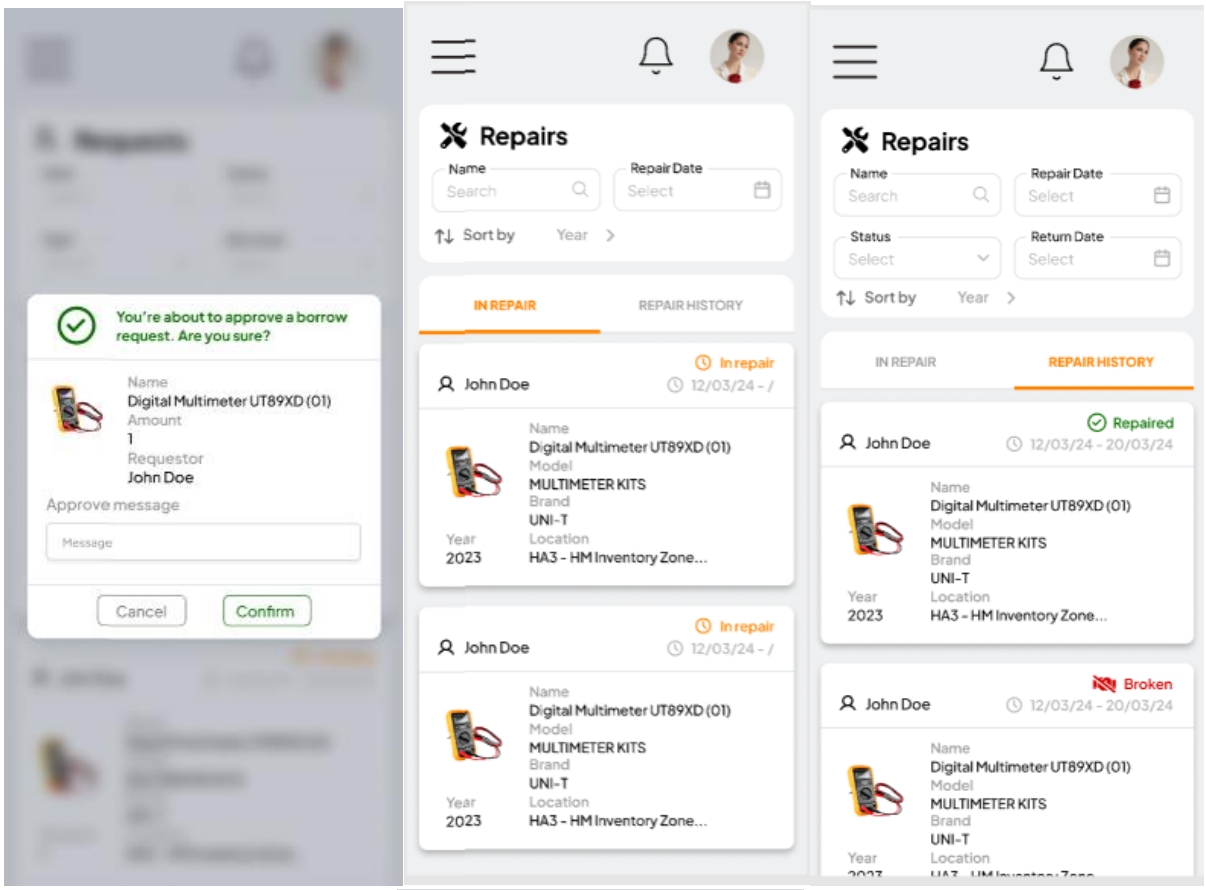
10.2.2. Student login:

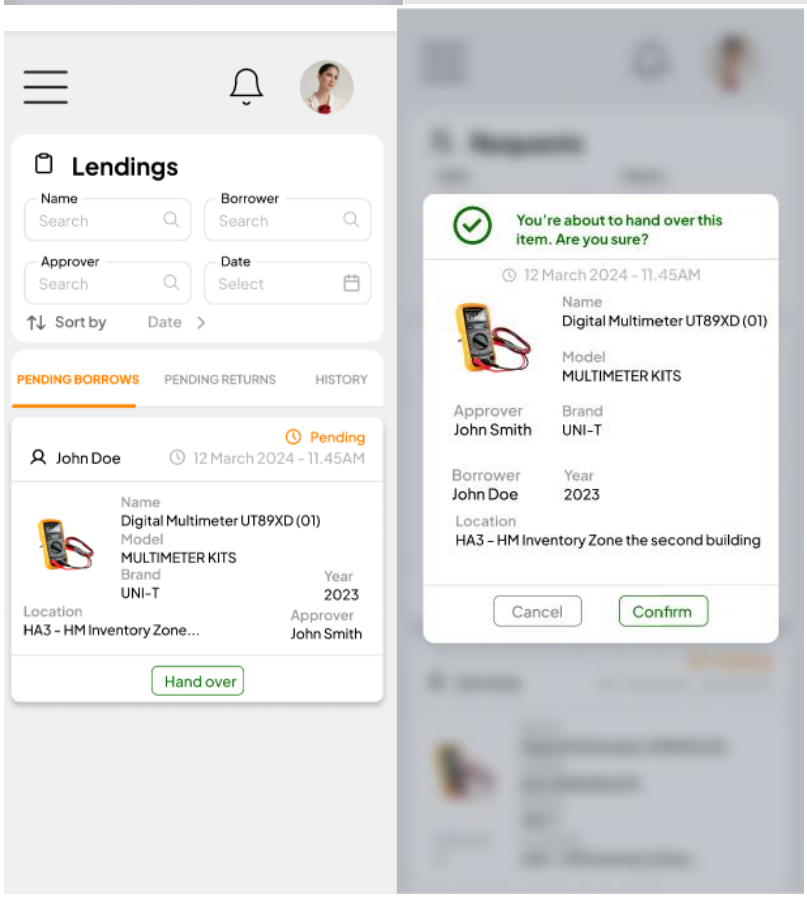


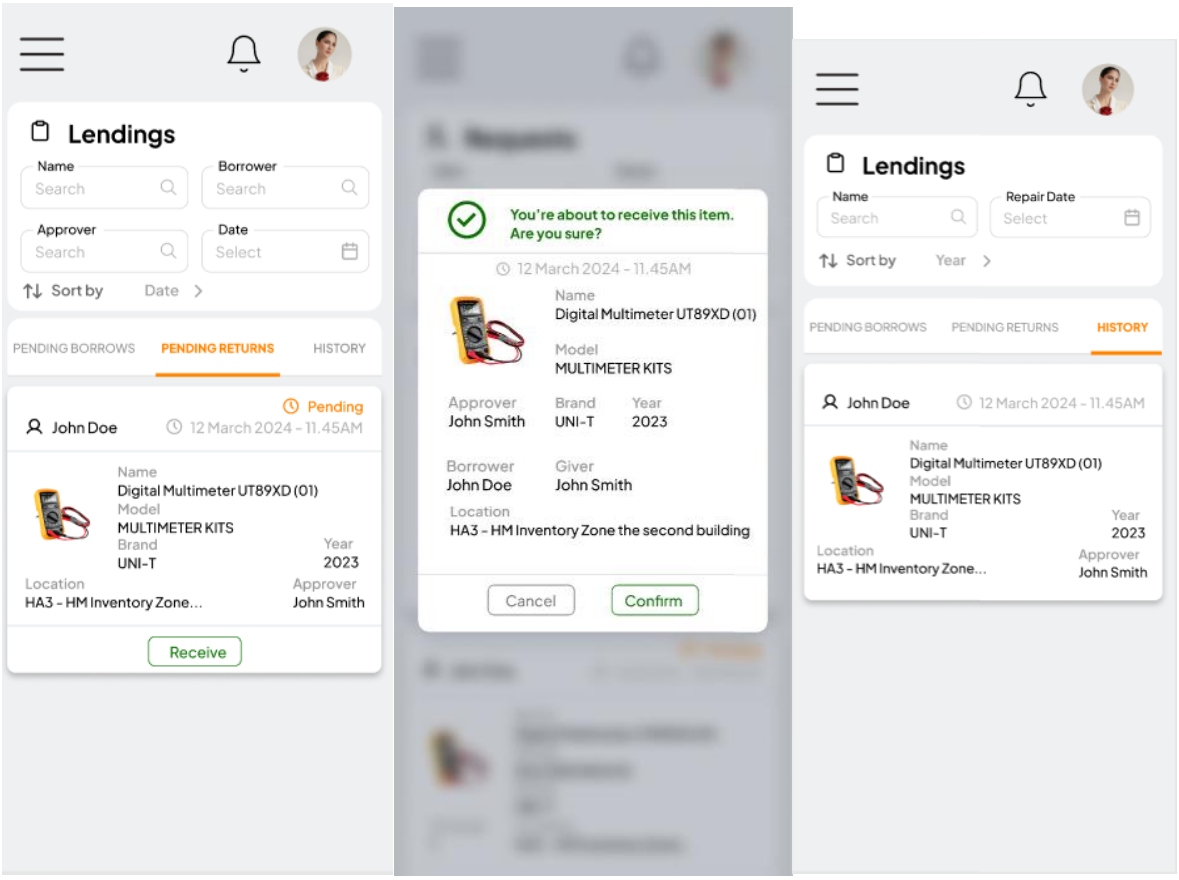


10.2.3. Supervisor login:

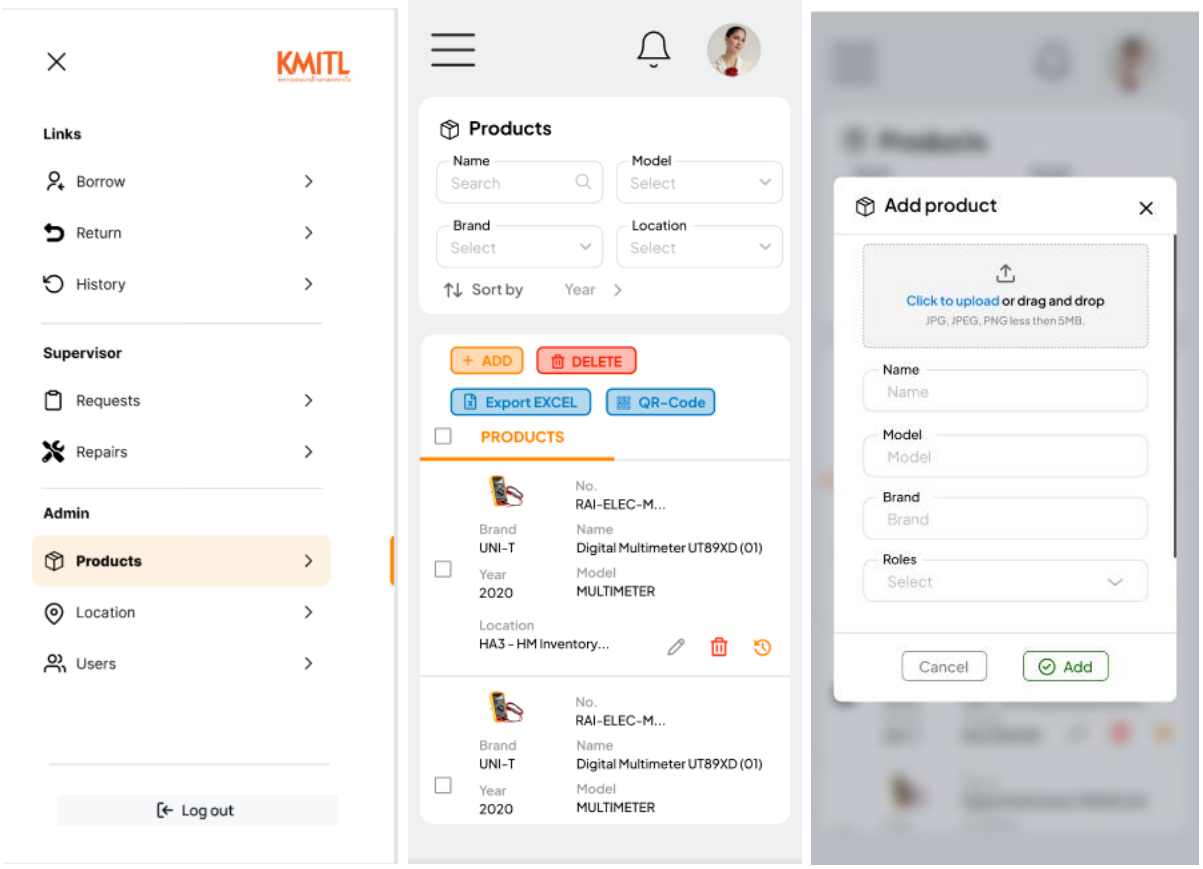


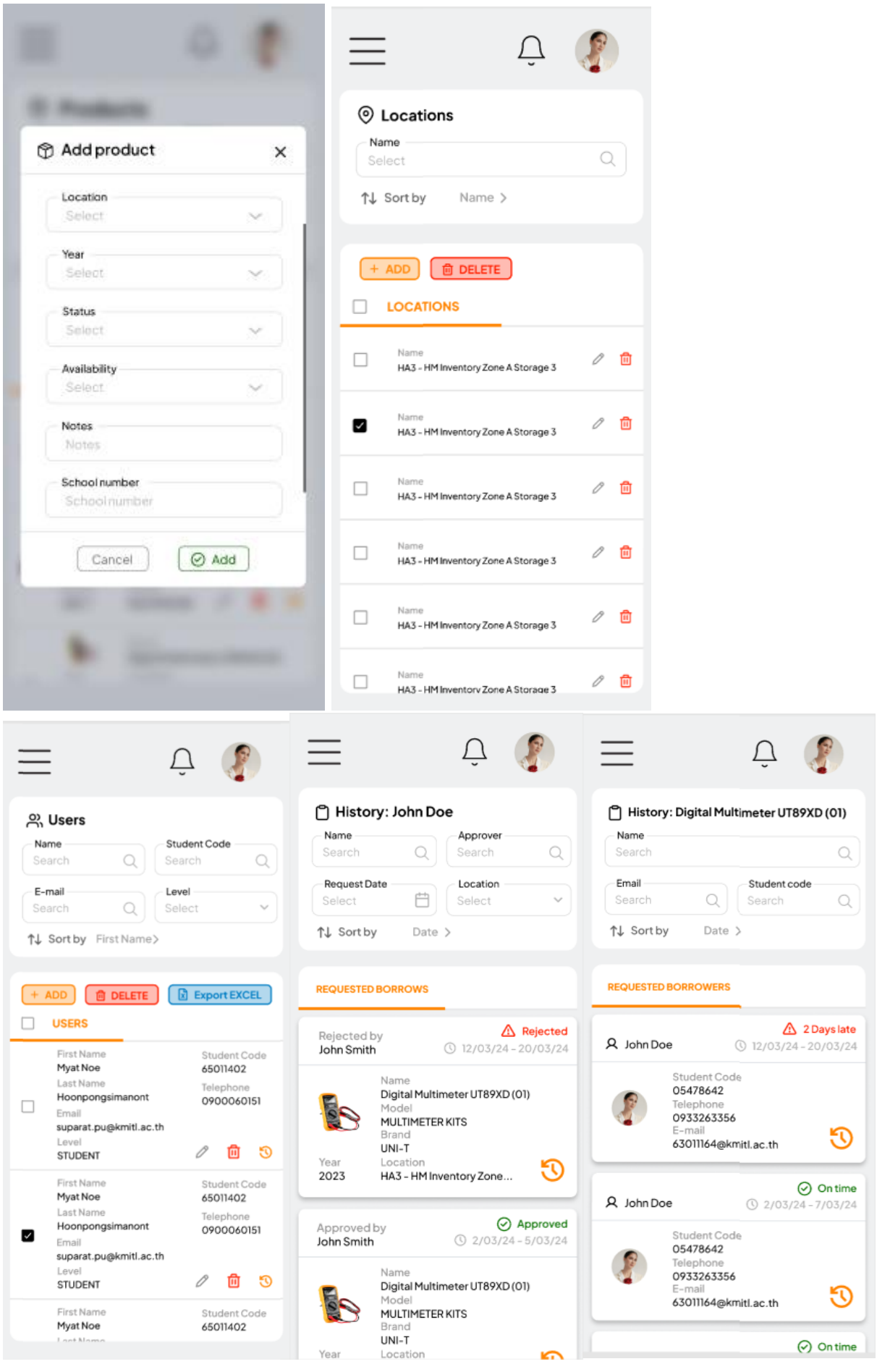

You're about to approve a borrow request. Are you sure?
Name
Digital Multimeter UT89XD (01)
Amount
1
Requestor
John Doe
Approve message
Message
Cancel
Confirm
Repairs
Name
Search
Repair Date
Select
Sort by
Year
IN REPAIR
REPAIR HISTORY
John Doe
In repair
12/03/24 - /
Name
Digital Multimeter UT89XD (01)
Model
MULTIMETER KITS
Brand
UNI-T
Year
2023
Location
HA3 - HM Inventory Zone...
Repairs
Name
Search
Repair Date
Select
Status
Select
Return Date
Select
Sort by
Year
IN REPAIR
REPAIR HISTORY
John Doe
Repaired
12/03/24 - 20/03/24
Name
Digital Multimeter UT89XD (01)
Model
MULTIMETER KITS
Brand
UNI-T
Year
2023
Location
HA3 - HM Inventory Zone...
John Doe
Broken
12/03/24 - 20/03/24
Name
Digital Multimeter UT89XD (01)
Model
MULTIMETER KITS
Brand
UNI-T
Year
Location


Lendings
Name
Search
Borrower
Search
Approver
Search
Date
Select
Sort by
Date
PENDING BORROWS
PENDING RETURNS
HISTORY
John Doe
Pending
12 March 2024 - 11.45AM
Name
Digital Multimeter UT89XD (01)
Model
MULTIMETER KITS
Brand
UNI-T
Year
2023
Location
HA3 - HM Inventory Zone...
Approver
John Smith
Hand over
You're about to hand over this item. Are you sure?
12 March 2024 - 11.45AM
Name
Digital Multimeter UT89XD (01)
Model
MULTIMETER KITS
Approver
John Smith
Brand
UNI-T
Borrower
John Doe
Year
2023
Location
HA3 - HM Inventory Zone the second building
Cancel
Confirm



10.2.4. Admin login:




Add product
Location
Select
Year
Select
Status
Select
Availability
Select
Notes
Notes
School number
School number
Cancel
Add
Locations
Name
Select
Sort by
Name
ADD
DELETE
LOCATIONS
Name
HA3 - HM Inventory Zone A Storage 3
Users
Name
Search
Student Code
Search
E-mail
Search
Level
Select
Sort by
First Name
ADD
DELETE
Export EXCEL
USERS
First Name
Myat Noe
Student Code
65011402
Last Name
Hoonpongsimanont
Telephone
0900060151
Email
suparat.pu@kmitl.ac.th
Level
STUDENT
History: John Doe
Name
Search
Approver
Search
Request Date
Select
Location
Select
Sort by
Date
REQUESTED BORROWS
Rejected by
John Smith
Rejected
12/03/24 - 20/03/24
Name
Digital Multimeter UT89XD (01)
Model
MULTIMETER KITS
Brand
UNI-T
Year
2023
Location
HA3 - HM Inventory Zone...
Approved by
John Smith
Approved
2/03/24 - 5/03/24
History: Digital Multimeter UT89XD (01)
Name
Search
Email
Search
Student code
Search
Sort by
Date
REQUESTED BORROWERS
John Doe
2 Days late
12/03/24 - 20/03/24
Student Code
05478642
Telephone
0933263356
E-mail
63011164@kmitl.ac.th
John Doe
On time
2/03/24 - 7/03/24
On time

10.3. Links to screens

Pc: https://www.figma.com/file/rAbnROflHkCGDIqVOcOKpa/Inventory-system-desktop?type=design&mode=design&t=vOCWCuKTRJqVi1gD-1

Mobile: https://www.figma.com/file/acxboRAx3vkOi3YaA4kCX8/Inventory-system-mobile?type=design&mode=design&t=vOCWCuKTRJqVi1gD-1

11. Conceptualization of the Application

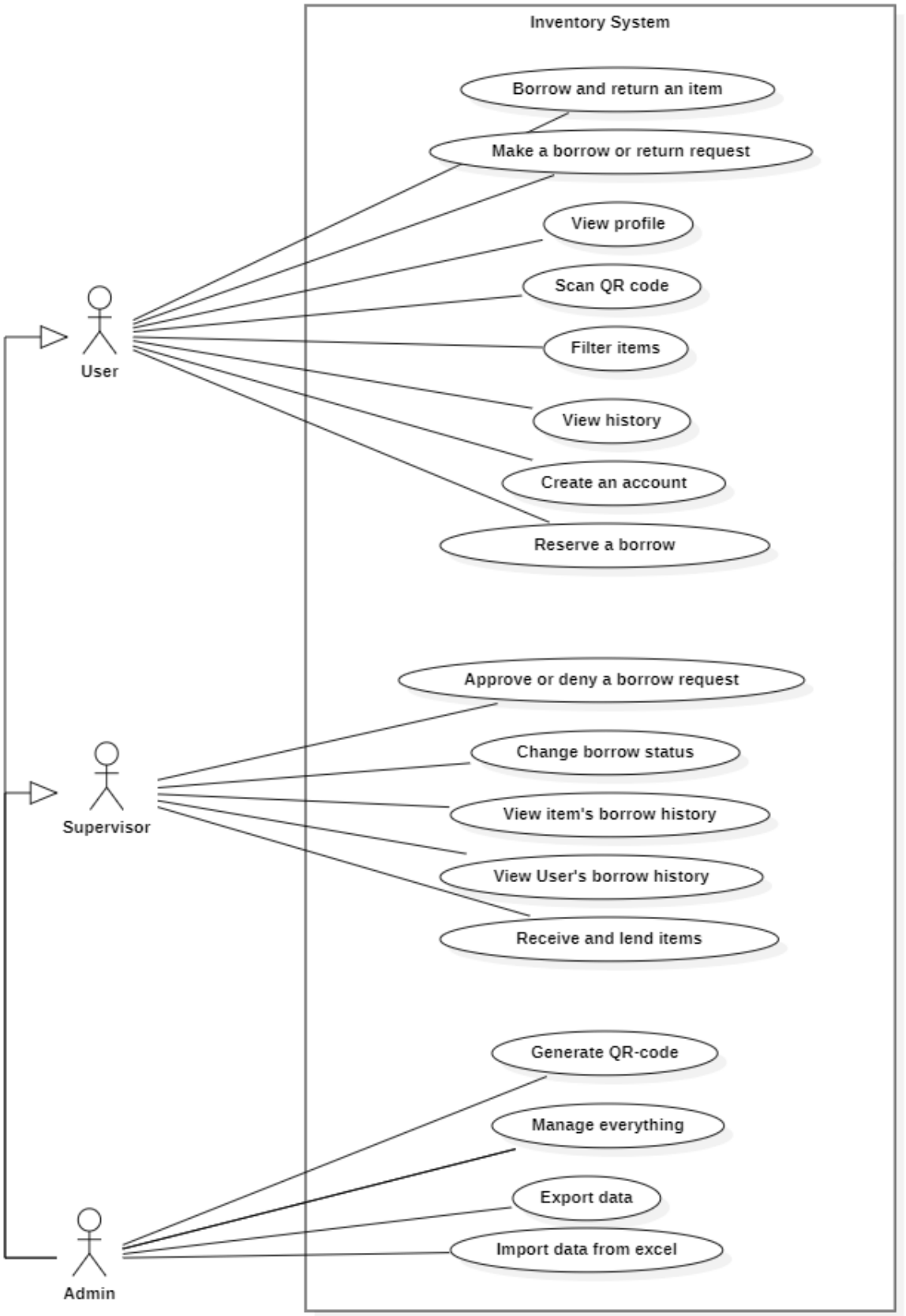
11.1. Deliverables

We discussed that at the end we must give a project handover. This will be two different files. One for the user on how to navigate and use the application and the other for developers to understand the code and the tools we used. Basically two manuals as project handover including our code.

11.2. Use case diagram

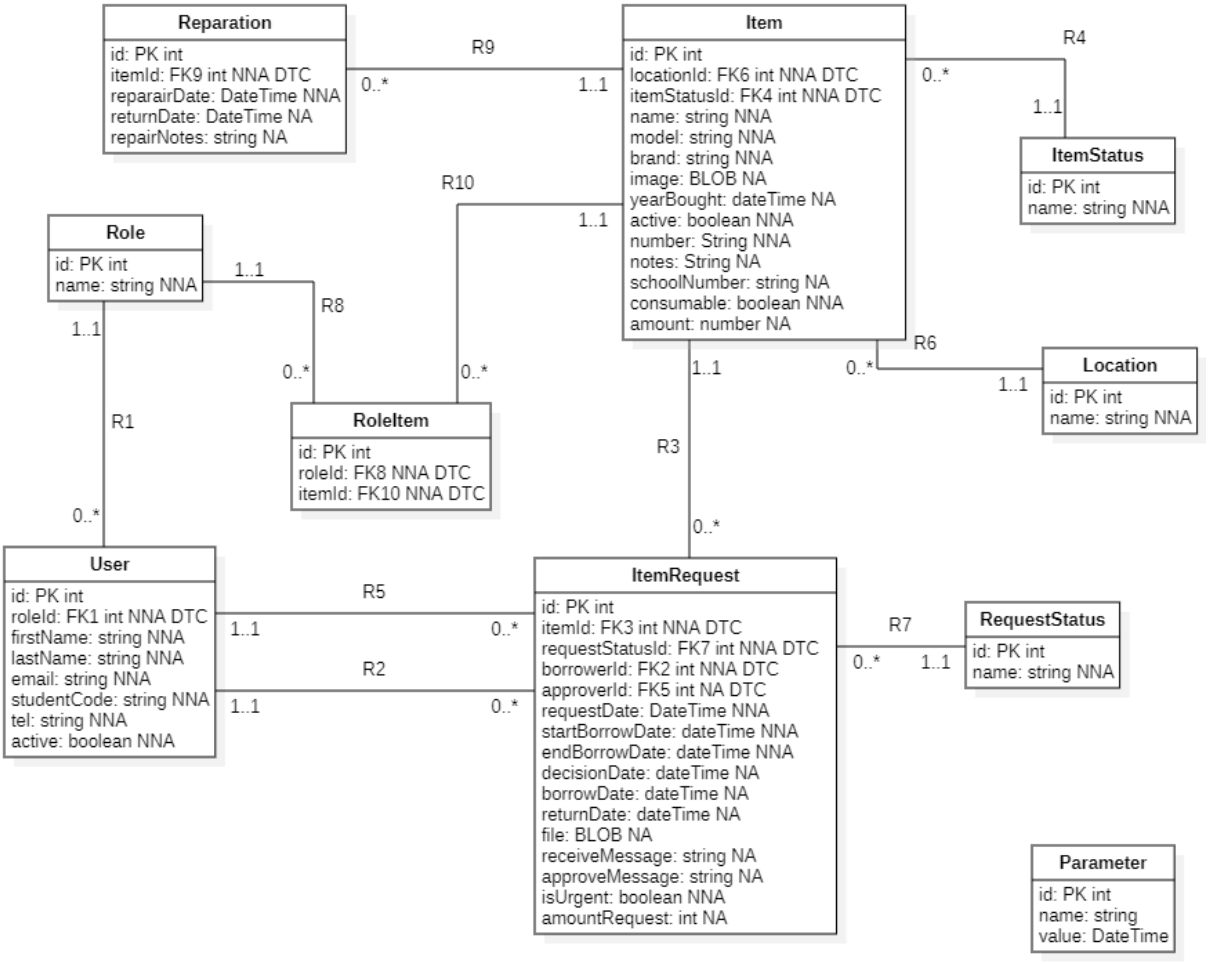
First, we will discuss our use case diagram. This is a visual representation of everything that a user should be able to do within our application.

Here you can see we have an actor called “user”. This can be a student or lecturer. We also have an actor called “supervisor”, he can do everything a user can and more. As last we have an actor called “admin”, he can do everything a user and a supervisor can and more.


Inventory System
Borrow and return an item
Make a borrow or return request
View profile
Scan QR code
Filter items
View history
Create an account
Reserve a borrow
User
Approve or deny a borrow request
Change borrow status
View item's borrow history
View User's borrow history
Receive and lend items
Supervisor
Generate QR-code
Manage everything
Export data
Import data from excel
Admin

11.3. Datamodel

We will now be discussing what structure we have come up with for the database, using a data model.



11.4. General

As a font we'll be using: Source Sans 3

And as a color schema we chose some colors that match the school:

- Text: #130E01 (black)
- Background: #FFFFFF (white)
- Accent: #CF4307
- Primary: #FF8400
- Secondary: #FFF5D6 (light yellow-ish)

12. References

Austin, D. D. (2023, October 30). *Medium*. Retrieved from Why Tailwind CSS Is 48% Better For Performance Than CSS-In-JS: https://medium.com/coding-at-dawn/why-tailwind-css-is-48-better-for-performance-than-css-in-js-93c3f9fd59b1#:~:text=It's%20an%20undeniable%20fact%20that,Block%2DElement%2DModifier).

Chaudhari, M. (2023, July 4). *Why should you use TypeScript for Next.js?* Retrieved from Medium: https://mayank1513.medium.com/why-should-you-use-typescript-for-next-js-7f0698ed8b2

kmitl. (n.d.). *about kmitl*. Retrieved from kmitl: https://www.kmitl.ac.th/

Osnat, R. (2020, 12 23). *Kubernetes Security Basics and 10 Essential Best Practices*. Retrieved from Cloud native academy: https://www.aquasec.com/cloud-native-academy/kubernetes-in-production/kubernetes-security-best-practices-10-steps-to-securing-k8s/

Rauch, G. (n.d.). *The easiest way to work*. Retrieved from Prisma.io: https://www.prisma.io/nextjs

Salman. (2023, October 25). *Innovating with Next.js: JavaScript vs. TypeScript — A Developer's Guide*. Retrieved from Medium: https://medium.com/@salluarsh/whether-to-use-javascript-or-typescript-to-build-next-js-af6f8b7ad4f3#:~:text=I%20personally%20recommend%20using%20TypeScript,maintainable%20and%20bug%2Dfree%20code.

Vulnerability Scanning Tools. (n.d.). Retrieved from owasp: https://owasp.org/www-community/Vulnerability_Scanning_Tools