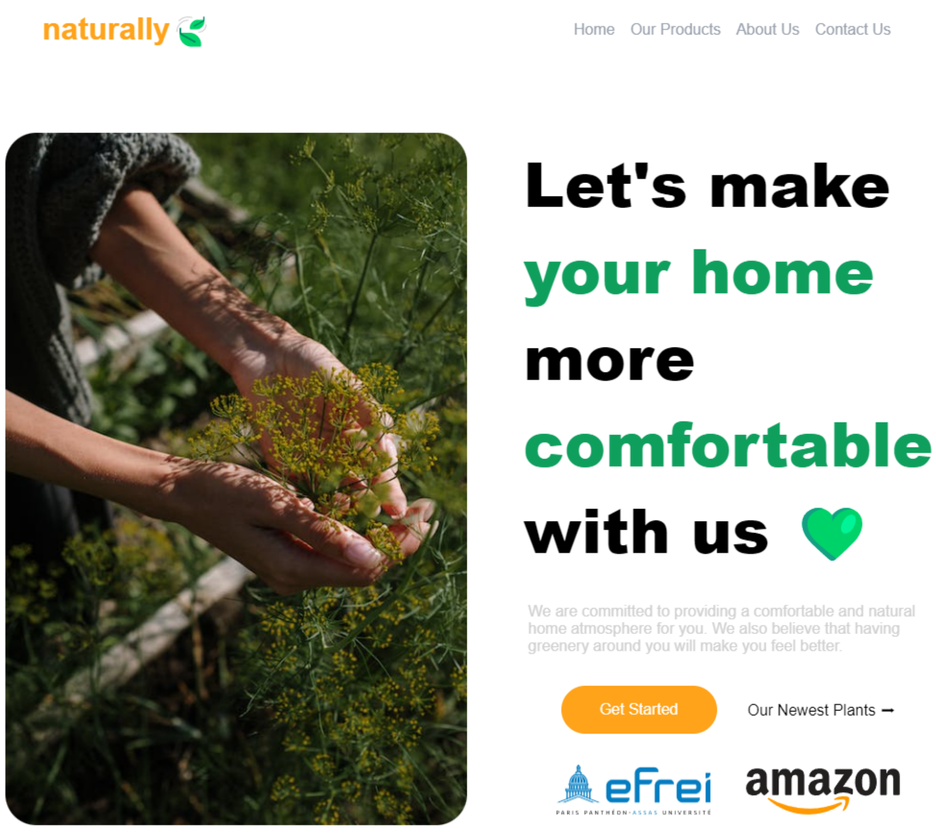
Plant Shop - Naturally


naturally
Home
Our Products
About Us
Contact Us
Let's make your home more comfortable with us
We are committed to providing a comfortable and natural home atmosphere for you. We also believe that having greenery around you will make you feel better.
Get Started
Our Newest Plants
efrei
amazon

Kévin Tran & Vincent LAM

2022-2023
To the attention of Mr. RAKOTONARIVO Rado
EFREI, L2 INT4

Summary

Introduction:	**2**
Features:	**2**
Implementation:	**2**
index.html	2
our-products.html	3
our-products.js	3
Conclusion:	**4**
Some pictures on mobile:	**4**

Introduction:

The application is a plant shop where users purchase various plants online.
We deployed the website with Github Pages using the following url:
https://lam-vincent.github.io/Plant-Shop-Naturally/html/index.html

Features:

The code provides a table containing information on various plants, including their names, prices and stock quantities. It also includes a form that allows users to add plants to their basket. Added plants are stored in the browser's localStorage, which guarantees data persistence even if the user closes the tab or window. This differs from sessionStorage, where data is lost when the tab is closed.

One of the main features is that when the user completes the purchase process, the application formats an e-mail containing all the information on the form and sends it to a fake company email address. This feature enables the application to function as a real online plant store and place an order.

In addition, the code incorporates a visually appealing confetti animation to produce a celebratory effect. The confetti animation is displayed both when the user adds an item to their basket and when they complete their order, creating a pleasant experience for the user.

As for the "mobile-friendly" feature, depending on the device used (Android phone, iOS phone and different browsers DevTools), it may not work the same way, although on my phone, mobile compatibility works perfectly.

We tried to use the best practices we'd read about on the internet, such as css names, creating a single object that is sent only once to the localStorage and placing files in organized folders.

And of course, we met all the requirements (or at least, we hope so) set out in the specifications, including the four pages, css, scripts, table, mobile-friendly mode, about us page and landing page, in which we followed an ambitious design found on Dribbble.

Implementation:

We're not going to talk about everything since you said we should only write 4 pages. We're just going to talk about our product page and how the backend works for that page. I don't even think we'll have time to talk about everything on this page, but hopefully the code will speak for itself and fill in what I haven't said.

index.html

We've put in the Efrei and Amazon logos, which are clickable and will open a new tab with the Efrei and Amazon websites respectively.

our-products.html

First of all, may I say that adding emoji to the table was such a brilliant touch. It's so pleasing to the eyes. I just love it.

Below the table, there is a form allowing users to select a plant and quantity, and then add it to their basket. The form includes an "ADD TO BASKET" button that triggers the addToBasket() function in which we add confetti using the JSConfetti library. The confetti animation is not activated when the quantity requested by the user exceeds the quantity in stock because of a simple return (line 10) in addToBasket() in our-product.js.

The <div> with the class "basket" represents the user's basket. It initially displays the basket header, including the basket heading and a "Clear All" button that clears the localStorage. The items added to the basket dynamically appear in the "basket-items" <div>, and the total price is updated accordingly.

Finalize the Order form allows users to finalize their order by providing their name, address, delivery date, and preferred delivery time. The form includes a "FINISH THE ORDER" button that triggers the sendOrder() function in which we add confetti using the JSConfetti library.

As an aside, I feel I should have used ids and not classes at times, as I only use some classes once sometimes. However, if you want to work with reusable classes or with reusable React components for example, or if you like the class system like with Tailwind CSS, I feel that the error is negligible compared to what I've learned about good practices, even though what I've learned is quite small. However, it's still an inaccuracy, but I have to say that ids are also used to retrieve values from form inputs, so I'd tend to write classes for things that could be reused and ids for data. This is just my humble opinion, if I may. But I'm still in the wrong though, I suppose.

our-products.js

addToBasket() is called when the user wants to add an item to the basket. It retrieves the selected plant name and quantity from the input fields, checks if the quantity exceeds the available stock, and then either updates the quantity of an existing item in the basket or creates a new basket item. It also updates the stock in the table, calculates the total price, adds confetti using the JSConfetti library and stores the basket and stock in the localStorage.

calculateTotalPrice() calculates the total price of all the items in the basket. It iterates over the basket items, retrieves the plant name and quantity for each item, calculates the item price and adds it to the total price. The function updates the total price element in the HTML and stores the total price in the localStorage.

getPlantPrice(plantName) retrieves the price of a plant based on its name. It uses a predefined object plantPrices to map plant names to their prices. If the plant name is not found, it returns 0.

getStock(plantName) retrieves the stock quantity of a plant from the table. It searches for a row in the table that matches the plant name and returns the stock quantity. If the plant name is not found, it returns 0.

updateStock(plantName, quantity) updates the stock quantity of a plant in the table. It searches for a row that matches the plant name and updates the stock value by subtracting the specified quantity.

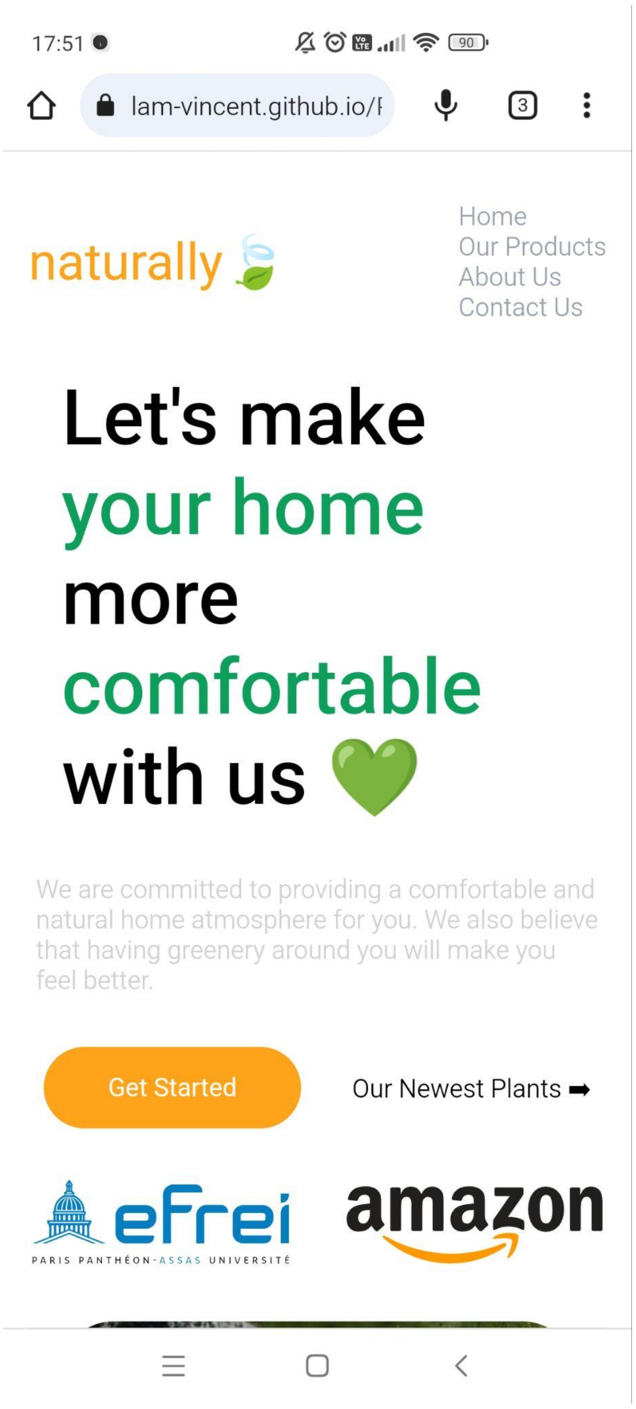
storeBasketAndStockInLocalStorage() stores the basket items and stock quantities in the localStorage. It retrieves the basket items from the HTML, converts them into an array of objects containing plant names and quantities. It also retrieves the stock information from the table and stores both the basket and stock data as an object in the localStorage.

retrieveBasketAndStockFromLocalStorage() retrieves the basket items and stock quantities from the localStorage and updates the HTML accordingly. It retrieves the data object from the localStorage, checks if it exists, and then recreates the basket items and updates the stock information in the table based on the retrieved data. It also calls calculateTotalPrice() to update the total price.

Conclusion:

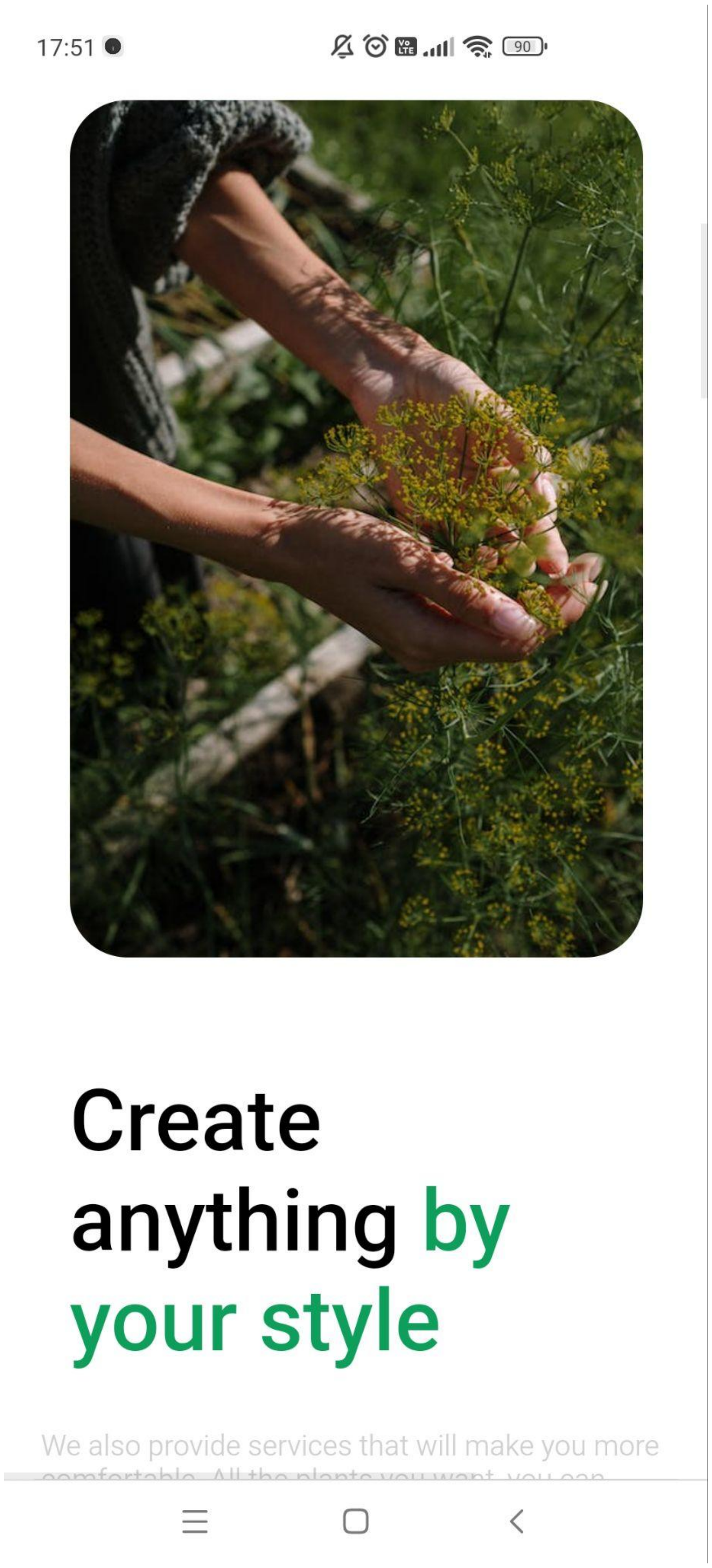
We were really invested in this project and we did our absolute best. Even if this application isn't perfect, we were very passionate about it. We're very happy with the result and whatever the grade, we feel like we've learned a lot through this project. Thank you for your time in reading this report and the code.

Some pictures on mobile:



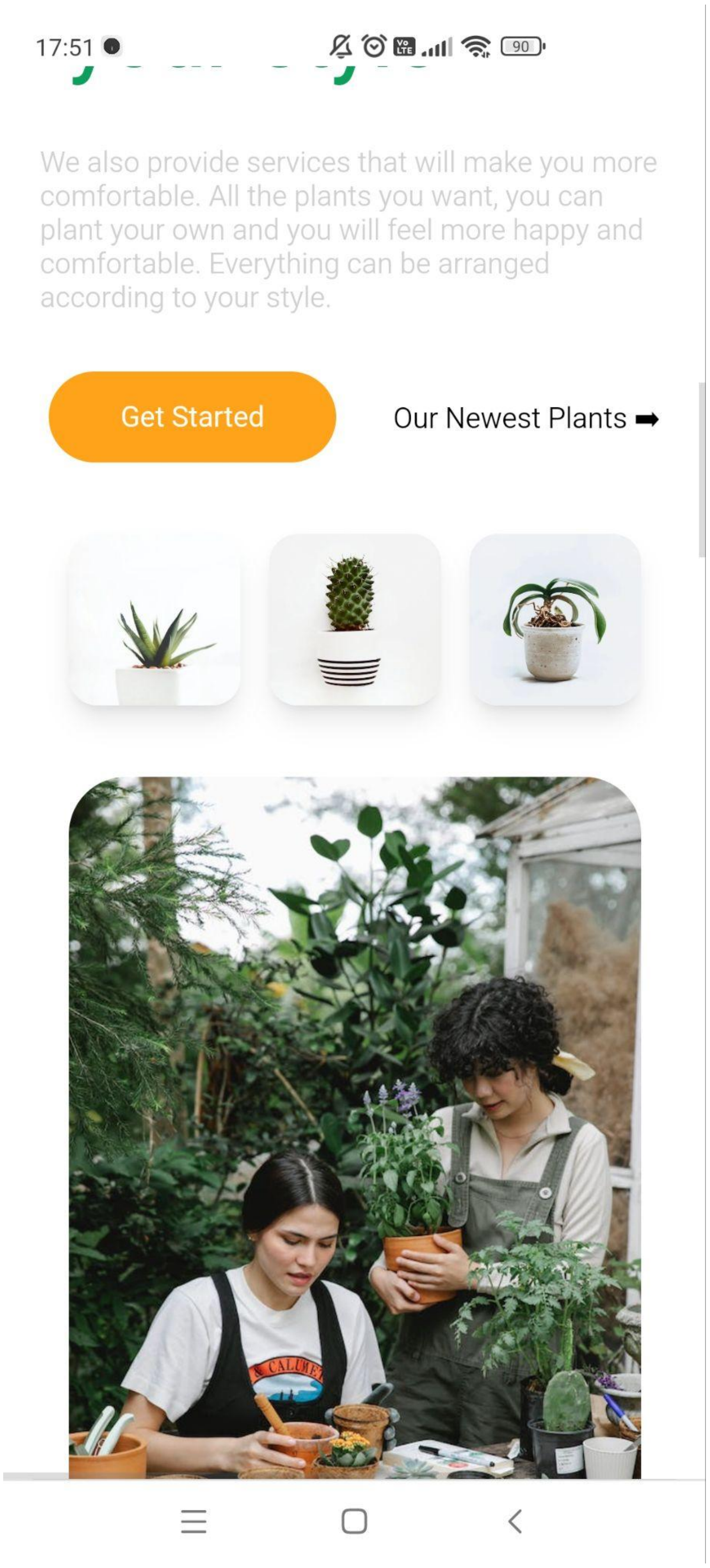
Let's make your home more comfortable with us 💚

We are committed to providing a comfortable and natural home atmosphere for you. We also believe that having greenery around you will make you feel better.

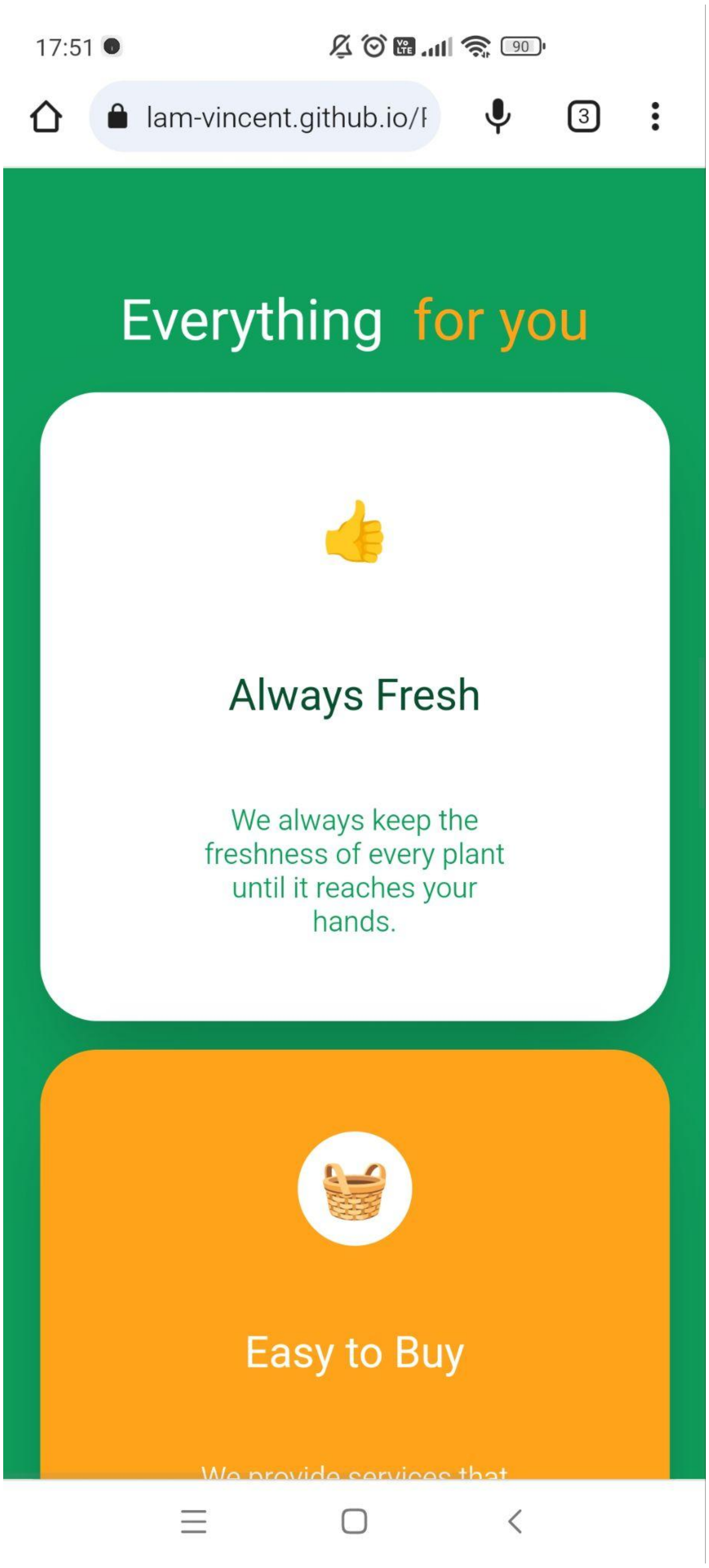


Create anything by your style

We also provide services that will make you more
comfortable. All the plants you want, you can


17:51

We also provide services that will make you more comfortable. All the plants you want, you can plant your own and you will feel more happy and comfortable. Everything can be arranged according to your style.



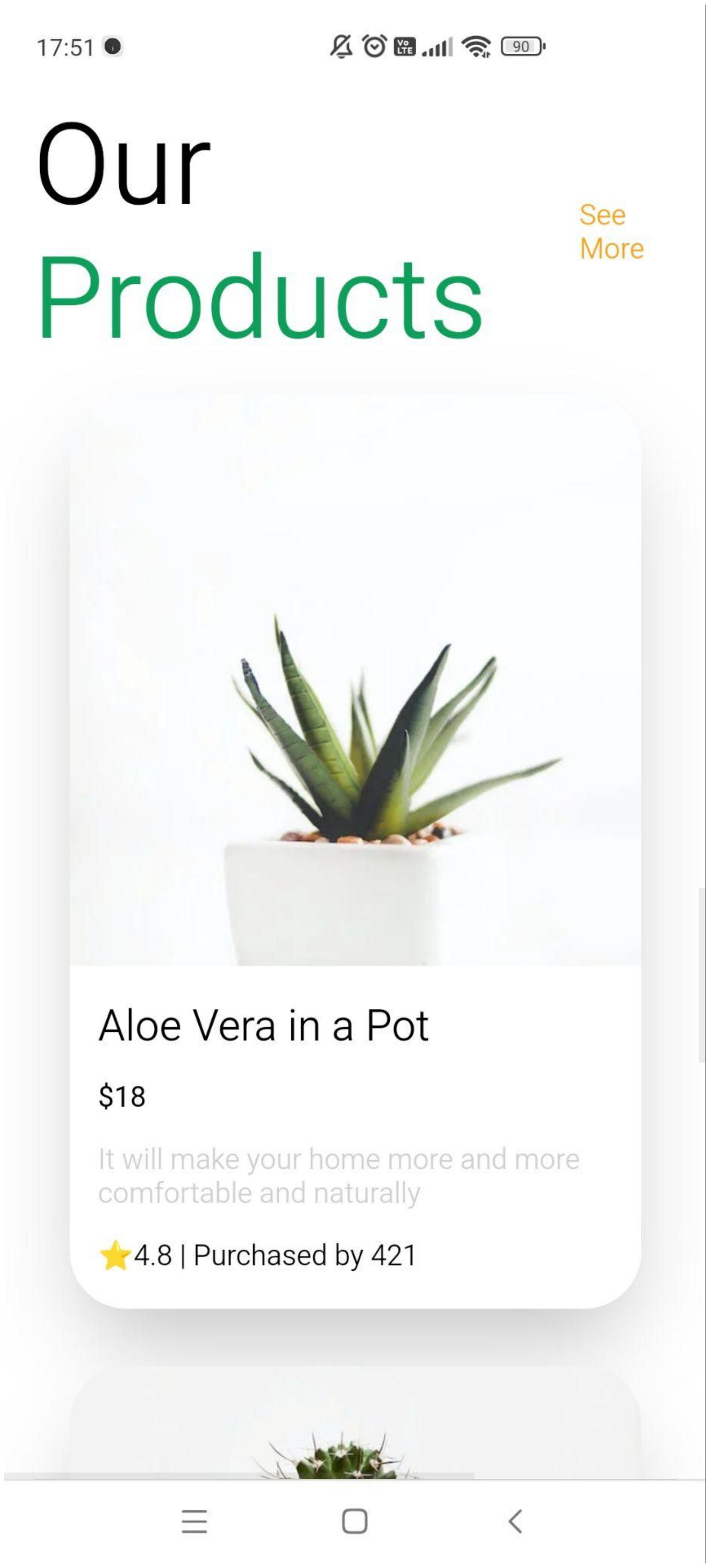
Everything for you

Always Fresh

We always keep the freshness of every plant until it reaches your hands.

Easy to Buy

We provide services that



17:51

Our

Products

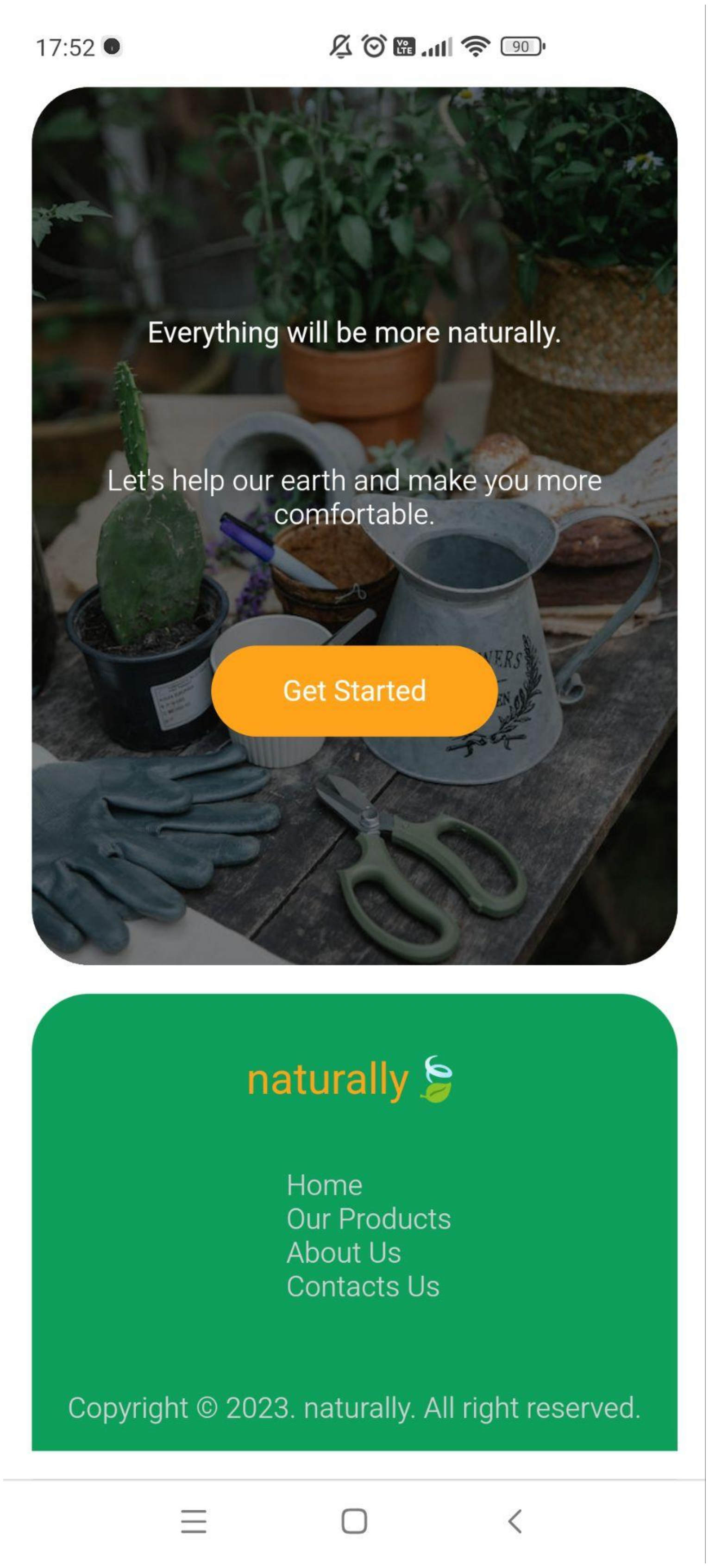
See More

Aloe Vera in a Pot

$18

It will make your home more and more comfortable and naturally

⭐4.8 | Purchased by 421


17:52
90
Everything will be more naturally.
Let's help our earth and make you more comfortable.
Get Started
naturally
Home
Our Products
About Us
Contacts Us
Copyright © 2023. naturally. All right reserved.