# Drillshare

A Web Application By

Team Lead: Tyler Heslop
Scrum Master: Kevin Karmacharya
Front End Lead: Trevor Henrich
Back End Lead Saksham Manandhar
Github Master: Arsany Attalla

# What is Drillshare?

Drillshare is a peer to peer tool rental service for general contractors, home owners, or anyone in need of a specific tool set.

Similar to ride-share clients like Lyft and Uber, Drillshare shows up where legacy rental services fall short.

# What is Drillshare?

Drillshare is a peer to peer tool rental service for general contractors, home owners, or anyone in need of a specific tool set.

Similar to ride-share clients like Lyft and Uber, Drillshare shows up where legacy rental services fall short.

# Why Drillshare?

Handymen, whether they be professional contractors or homeowners, often suffer from limited stock at their local legacy rental business.

Having to arrive at a store just to leave empty handed can really be a bummer. If only you could rent a tool from someone in your neighborhood in a mutual agreement. How would you find such a deal?

# Why Drillshare?

To make matters worse, in 2020, the Covid-19 pandemic flipped the world on its head. With many people stuck working from home, repairs and construction rose.

With raw building materials becoming more expensive as a direct result, Drillshare aims to save the user money while providing income to tool owners.

# What is our tech stack?

Drillshare uses MongoDB, Express, and Node for the backend & ReactJS for the front end.

This stack, sometimes called the MERN stack, is among the most popular tech stacks being used today.

# Hosting

Our website is hosted on AWS T2 server. We used caddy to automatically setup our SSL. We chose it because it was easy to use compared to apache and came well recommended by the TA.

# React

React provides a flexible and high performing JS library that allows us to build single page applications that were responsive and fast.

React offers multiple ways to solve a web developers problems. We selected functional components rather than class based components because it was easy to pick up and simple to use.

# UI & UX

Material UI provides thoughtfully composed components that are customizable and reusable without having to write as much CSS.

Material UI uses grids based on flexbox allowing for a positive viewing experience on multiple screen sizes.

# UI & UX, cont'd

We chose MUI because of its robust costomizability. MUI has a theme object that can be modified with great simplicity to affect the global styling of the site.

The global theme object allowed for easy tweaking to the global styles while also allowing for local styles on specific pages.

# UI & UX, cont'd

MUI also offers a plethora of reusable components that allow for their own local styling beyond what the global styling allows.

This enabled us to send our CSS and other styles to our components like props, making for easy UI tweaks on the fly, to fit in well with Agile development.

# MongoDB

MongoDB is a NoSQL database which allows it to handle a high volume of data quickly with scalability in mind. The ability to update schemas easily creates a more developer friendly experience

# MongoDB, cont'd

Data is able to be stored in unstructured, semi-structured- or structured tables. Data can be stored in forms more similar to what the app will use.

# Node & Express

In order to maintain simplicity, we selected Nodejs and Express to serve the backend data to the front end. We wanted the backend and frontend to use the same language.

# P1 functions

Login/Registration
The user can register an account, login to that account, and add information to their profile.

Create Listing
The user can list an item for rent for others to discover.

Rent
A user can rent a listing.

# P1 functions

Return

A user can return a listing.

Delete

An owner can delete their listing

Search

A user can search for a specific item

# P2 functions

- Order History
- Messaging
- Filtering

# P3 functions

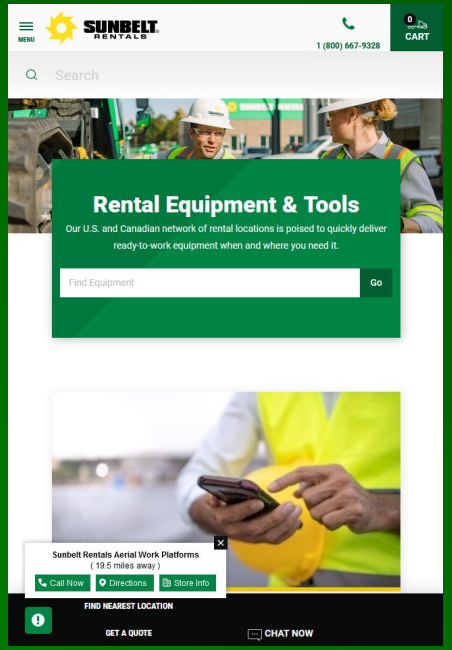
- Notifications
- Reviews
- Recommended Items

# P3 functions

- Notifications
- Reviews
- Recommended Items

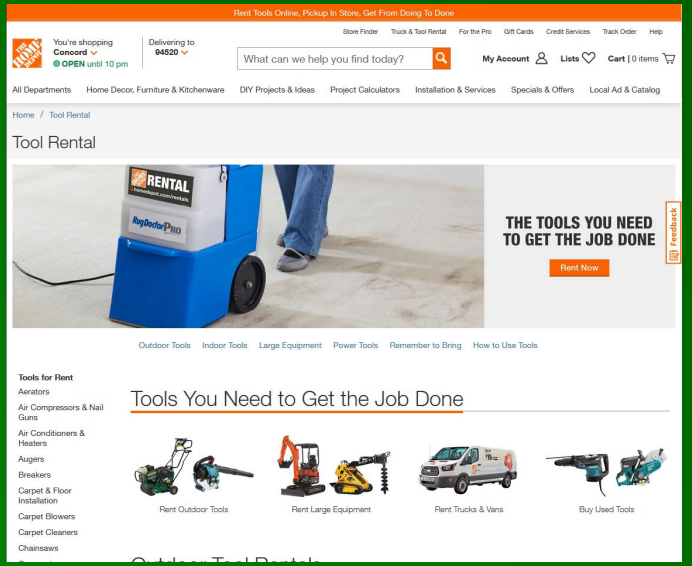
# Sunbelt Rental

- Cluttered
- Not intuitive to use
- Redundant
- Slow Loading



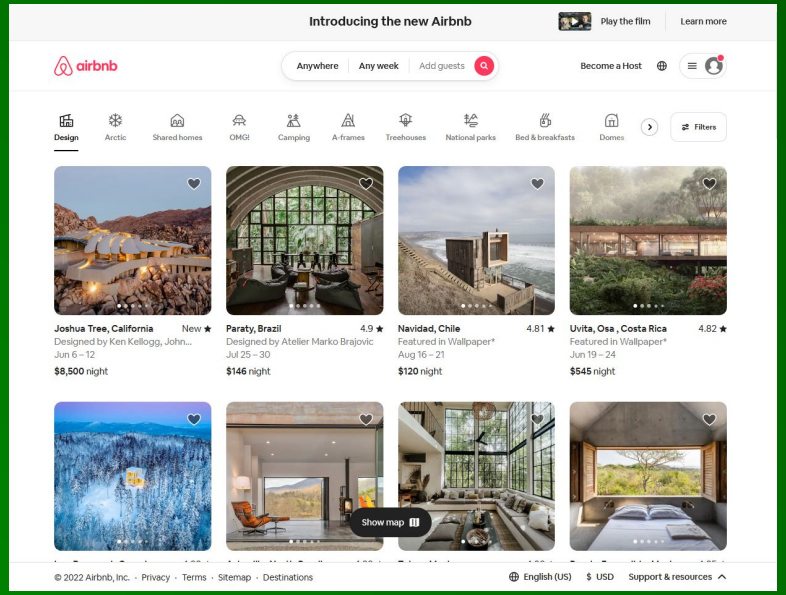
# Home Depot

- Cluttered
- Hard to use
- Overwhelming

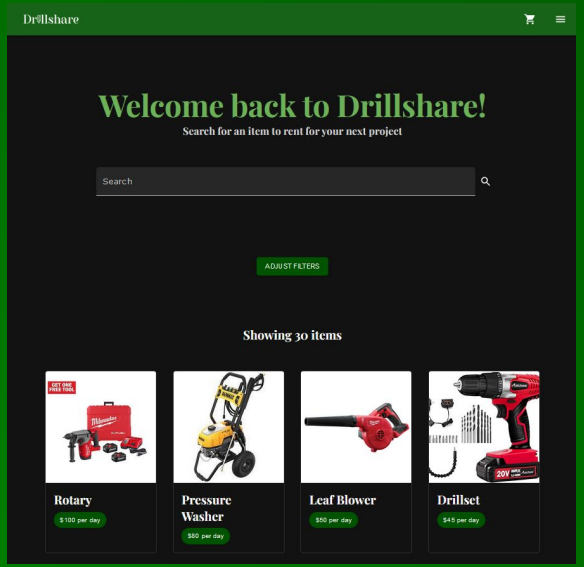


# Airbnb

- Simple
- Easy to use



# Demo


Drillshare
Welcome back to Drillshare!
Search for an item to rent for your next project
Search
ADJUST FILTERS
Showing 30 items
Rotary
$100 per day
Pressure Washer
$80 per day
Leaf Blower
$50 per day
Drillset
$45 per day

# Scrum Meetings

- We met twice a week, sometimes 3 times a week, where we would go over what needed to be done or if we needed to change gears.
- After meetings, we would sit in discord while writing code, making jokes, and helping eachother

# Styleguide

```
/**
 * Signup
 * @return {JSX.Element}
 * @constructor
 * @description
 * sign up for drillshare!
 */
```

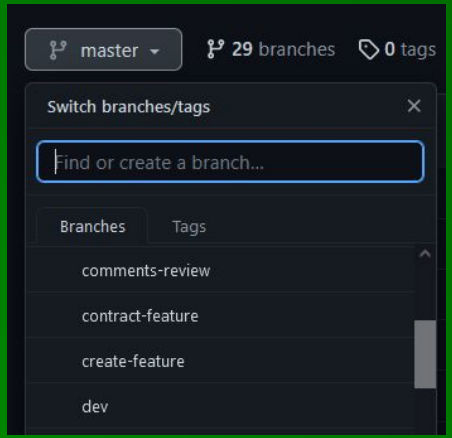
We’re using google styleguide which emphasizes readability and documentation above all. Were using JSDocs to further document our code.

# Code Review

When pushing code, we had a discord bot that would notify the server about git pushes and pull requests.

When a pull request was made, everyone was notified and would go to Github to leave a comment after reviewing the code.

# Code Review




We would start by branching off the dev branch. This feature branch would then be merged back to dev before dev was merged to master.

# Testing

Postman was used for testing endpoints and selenium was used for our integrated testing.

# Work Together

- The front end & back end met to collaborate on issues that plagued both teams.
- The different teams often met on their own to work together on similar issues

# Work Together

- The front end & back end met to collaborate on issues that plagued both teams.
- The different teams often met on their own to work together on similar issues

# Communication

A discord channel was set-up with various bots & channels to support the development of Drillshare.

We kept track of meetings in text channels where we shared the meeting minutes & any active todo lists

The frontend & backend had their own respective channels to communicate together

# Future Updates:

- A toggle for Light & Dark modes
- Return confirmation from Owner
- Recommendations
- Trending
- Order History
- Robust Delivery System

# Group Complications

- All 5 of us had very different schedules that made it difficult to conduct a full scrum meeting at times.
- Server Limitations (1GB ram server with 2GB node modules.