



A mobile phone shop designed and built by:

The Admirable Owls

Date: 10.06.2020



OUR PHONES
ARE GETTING
THINNER AND
SMARTER.
ARE YOU?

Elevator pitch

- Gives customers the flexibility of third party sim plans
- Adds competition to the big 3 telecommunication companies
- Give users a platform to pay for phones on a month basis

Concept

- A simple user friendly web page, that allows users to search or browse through a collection of phones where they can add to cart to be purchased.
- Motivation for development: We don't want our users to be restricted to the major companies and their limited and overpriced plans when purchasing a new phone.

User Story

- As a user, I want to buy a new phone without paying outright and also keeping my plan with a third party plan provider.
- I WANT to be able to search for a phone based on the price and brand, SO that I can see which phones match my criteria.
- WHEN I click on a mobile phone that i would like to buy, THEN I am presented with a selection of colours and features, SO that I can pick the one I would like.
- WHEN I select a colour and click Add to Cart, THEN the selected mobile phone will be added to my shopping cart.
- WHEN I'm ready to checkout, THEN I click on the checkout button to fill out my personal details and finalise my payment.

Process

- Technology used
- Breakdown of tasks and roles
- Challenges
- Successes

Technologies used

- ✓ Must use a Node and Express server
- ✓ Must use Handlebars.js as the template engine
- ✓ Must be backed by a MySQL database with a Sequelize ORM
- ✓ Must utilize both GET and POST routes for retrieving and adding new data
- ✓ Must be deployed using Heroku (with data)
- ✓ Must utilize at least one new library, package, or technology that we haven't discussed - (NPM Lazyload)
- ✓ Must have a polished front end/UI
- ✓ Must have a folder structure that meets the MVC paradigm
- ✓ Must meet good quality coding standards (indentation, scoping, naming)
- ✓ Must protect API keys in Node with environment variables

Breakdown of tasks and roles

1. Notes - mind mapping
2. Work packages into Google sheets
3. Work packages into GitHub Kanban
4. Monitoring progress from Kanban

Notes and mind mapping

```
*Work packages - Notepad
File Edit Format View Help

back end
- Set up file structure
- Write app.js and routes
- Create database
- Create tables/model
- populate db with a seed file
- code controller ORM sequelize
- write the Handlebars logic for outputting HTML

Front end
- Create simple Bootstrap front end with pagination
- display all products in cards
- click on a product to get detailed view (nice to have)
- add a product to cart
- remove product to cart
- go to check out page to get a detailed view of the cart
```

```
Store brief - Notepad
File Edit Format View Help

Design a basic web store, it should have items/orders and users. It sh ^
IT should be designed to use MVC
It should have a api
and if you have time a UI
- 1 point, design a basic schema (can be as orm models)
- 2 point create basic API
- 3 create basic UI
Should use a group git hub project/ that has a git ignore... sh

How do you interact with the store?

What type of store do we want?
Is this about a one to many DB relationship?
How many products?
What relationships might we find in the database?
How basic or advanced is this app?
How long have we got to finish it?
Do we need to mock up the front end before we start?
I think it should look good
Shall we use the DB tool Trent uses to design the schema?
How many products?

The shop
Mobile phone store with 6 phones and some accessories

What is inscope for this project?
What out of scope?

Core functionality from the database?
- Search phones based on different properties
- Add phones to a basket
- View the basket
- Take things out of the basket
- Suggest accessories/or add accessories
- Purchase history

Nice to have functionality
- Some comparison by attribute, bat life, weight, pixels
- Customer table
```

```
phone shop model - Notepad
File Edit Format View Help

Model name
Manufacturer
Price
Year of manufacture
Colour
Battery life
Display size
Camera pixel
Night mode

Customer first_name
Customer last_name
Email address
```


[illegible]

GitHub Kanban

The screenshot shows a web browser window displaying the GitHub Kanban board for 'amarr001 / Project_2'. The browser's address bar shows the URL 'github.com/amarr001/Project_2/projects/1'. The page header includes navigation links for 'Pull requests', 'Issues', 'Marketplace', and 'Explore'. The main content area shows the Kanban board with four columns: 'To do', 'In progress', 'Done', and 'Pinned for reference'. Each column contains several cards with task details, including titles, descriptions, assignees, and due dates. The 'To do' column has three cards, 'In progress' has three, 'Done' has three, and 'Pinned for reference' has three. The cards are organized into groups, with some having expandable sections for more details. The 'To do' column is currently selected, and the 'In progress' column is also visible. The 'Done' column shows completed tasks. The 'Pinned for reference' column contains links to external documents and a 'VIEW PROJECT PLAN HERE' link. The bottom of the page shows a search bar and a 'Filter cards' button.

lodafor | lodafor | Built-in | Sequeli | (5) Add | Sequeli | javascript | Inbox (5) | My Drive | Project | coding | Admiral | Project | Post Att | Proj x

github.com/amarr001/Project_2/projects/1

Apps | javascript - How ca... | (5) Create Partials -... | fetching all elem | Seq include | Sequelize relations... | Locafone Project | Project 2 | My Site | class | videojs/videojs: Vid...

Search or jump to... | Pull requests | Issues | Marketplace | Explore

amarr001 / Project_2

Watch 2 | Star 0 | Fork 0

<> Code | Issues 0 | Pull requests 0 | Actions | Projects 1 | Wiki | Security 0 | Insights

Project 2
Updated 4 days ago

Filter cards | + Add cards | Fullscreen | Menu

To do

- Environment variables for deployment!!!
Added by BenBugs
- Front end
 - Write Handlebars logic for front end based on database schema and wireframesWHO - Ana/Andy
DUE - 03.06
Added by BenBugs
- Back end
 - Select a list of 12 phones and add to Kanban card, please use Optus website, the images are betterWHO - Udara
DUE - 02.06
Added by BenBugs
- Documentation

In progress

- Documentation
 - Joins using sequelize and complementary work with Udara on database migrationsWHO - Ana
DUE - 29.05
Added by amarr001
- Backend
 - Write app.js and routesWHO - Andy
DUE - 30.05
Added by Undies
- Documentation
 - Spec out the technology to be used in the shopping cart and write bullet points on the approachWHO - Nathan
DUE - 01.06

Done

- Project Management
 - Research GitHub managementWHO - Udara/Ana
DUE - 28.05
Added by BenBugs
- Design
 - Define data fieldsWHO - All
DUE - 29.05
Added by BenBugs
- Design Work
 - Screens and responsive layout
Deliverable: .psd filesDUE - 30.05
WHO - Ben
Added by BenBugs

Pinned for reference

- ASSOCIATIONS
https://docs.google.com/spreadsheets/d/1ahalpTbyslszyDzLD_TrQfy2ayc6MhEbxVQ8tFu_3U/edit?usp=sharing
Added by BenBugs
- VIEW PROJECT PLAN HERE
<https://docs.google.com/spreadsheets/d/1wy6NnehOmYqelPdPRwO5UuZWlBb443w86qjZC1D8p0/edit?usp=sharing>
Added by BenBugs
- Documentation
 - GitHub good practices<https://docs.google.com/document/d/1zBHd8063mTii5ZApFjY1emVMgDdlMyixTbL75L804/edit?usp=sharing>
Link to a document containing simple instructions on how to work with Github branches

GitHub best practice

Working with branches in Git and Github:

Create your new branch: **git checkout -b BranchNameHere**

(this command creates the branch and automatically switches to it)

You can check how many branches your repository has and in which branch you are at (asterisk next to it) with the command: **git branch -a**

Push your branch to the remote repository:

git push -u origin feature_branch_name

To toggle back and forth between branches: **git checkout branchName**

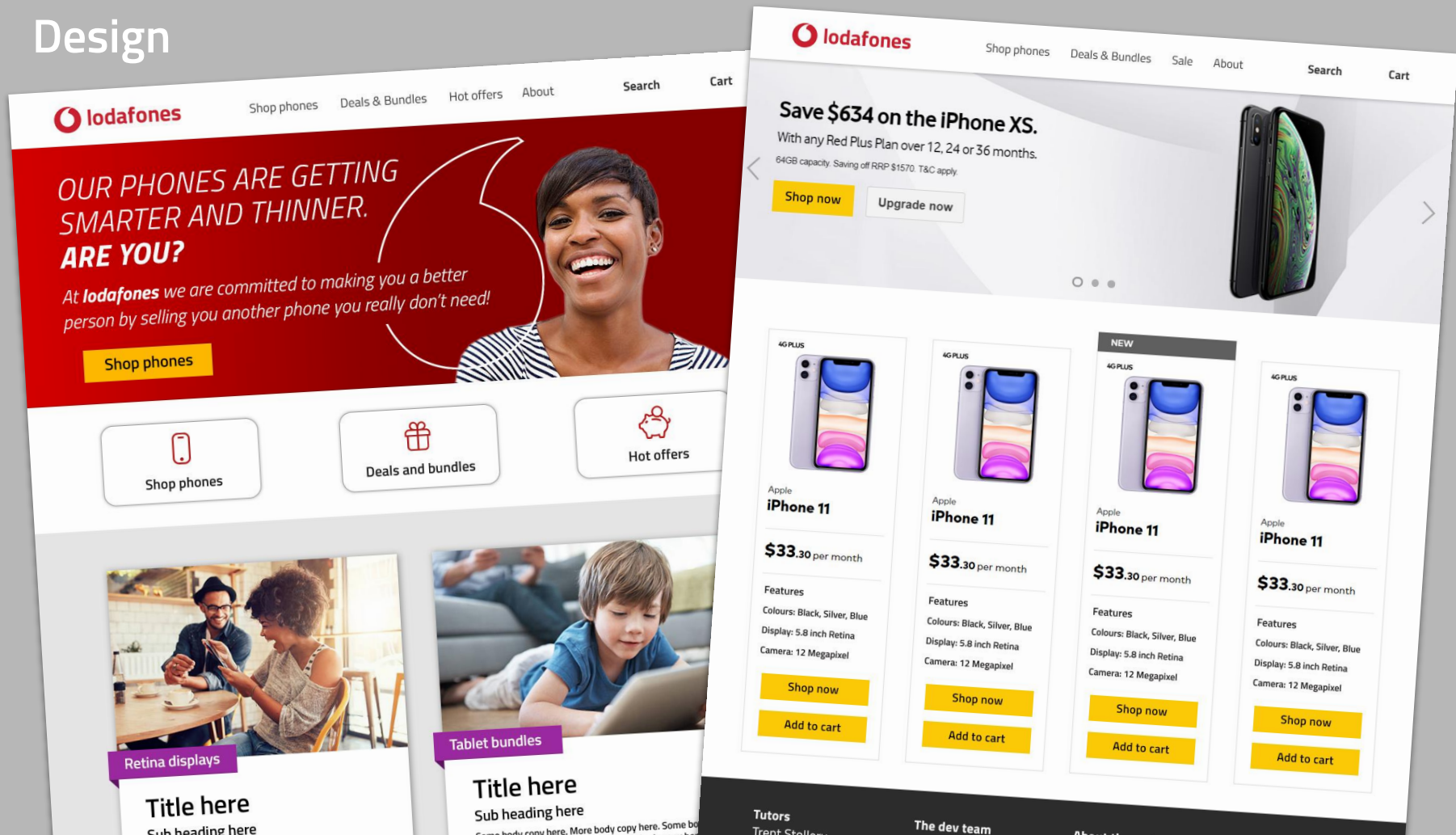
Make commits **at logical points** and name it so others can understand what you have done and why

When merging (moving code between branches) **we need to be on the branch that we want to merge to.**

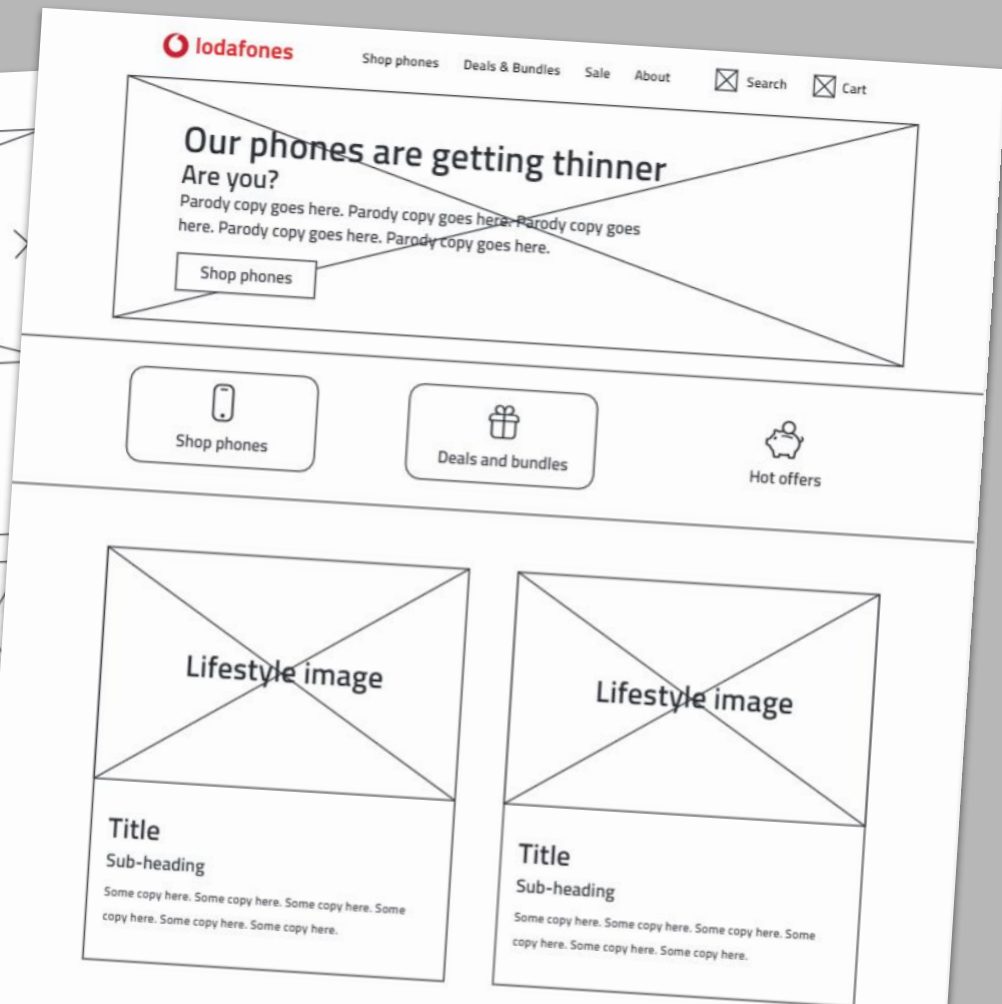
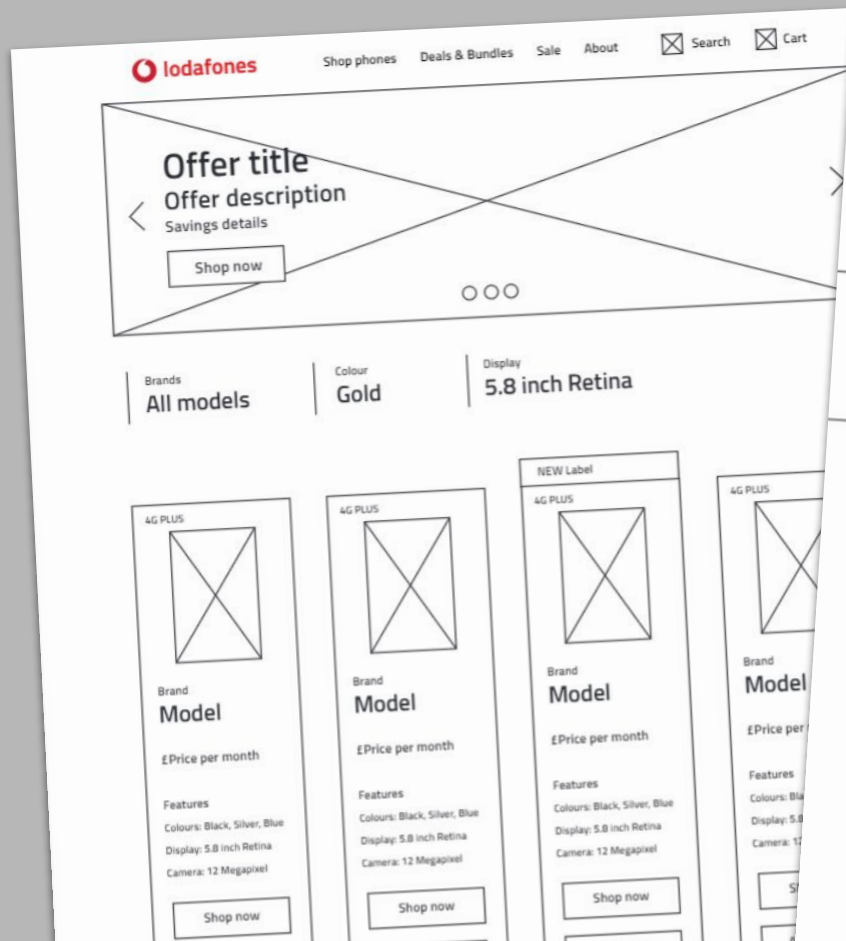
Once on the master branch (if this is the branch you want to merge your code to), run the merge command: **git merge branchName --no-ff** (the additional "--no-ff" tells git we want to retain all of the commit messages prior to merge, this makes tracking changes easier in the future). After merging, run **git push origin master**. If you could not complete the merge and receive this message: You have not concluded your merge (MERGE_HEAD exists), delete the merge by running: **git merge --abort** then **git pull**

Before making changes in your file within your branch, run a **git pull origin master** so you update your branch with the changes in the master branch. Do it every time a big change is

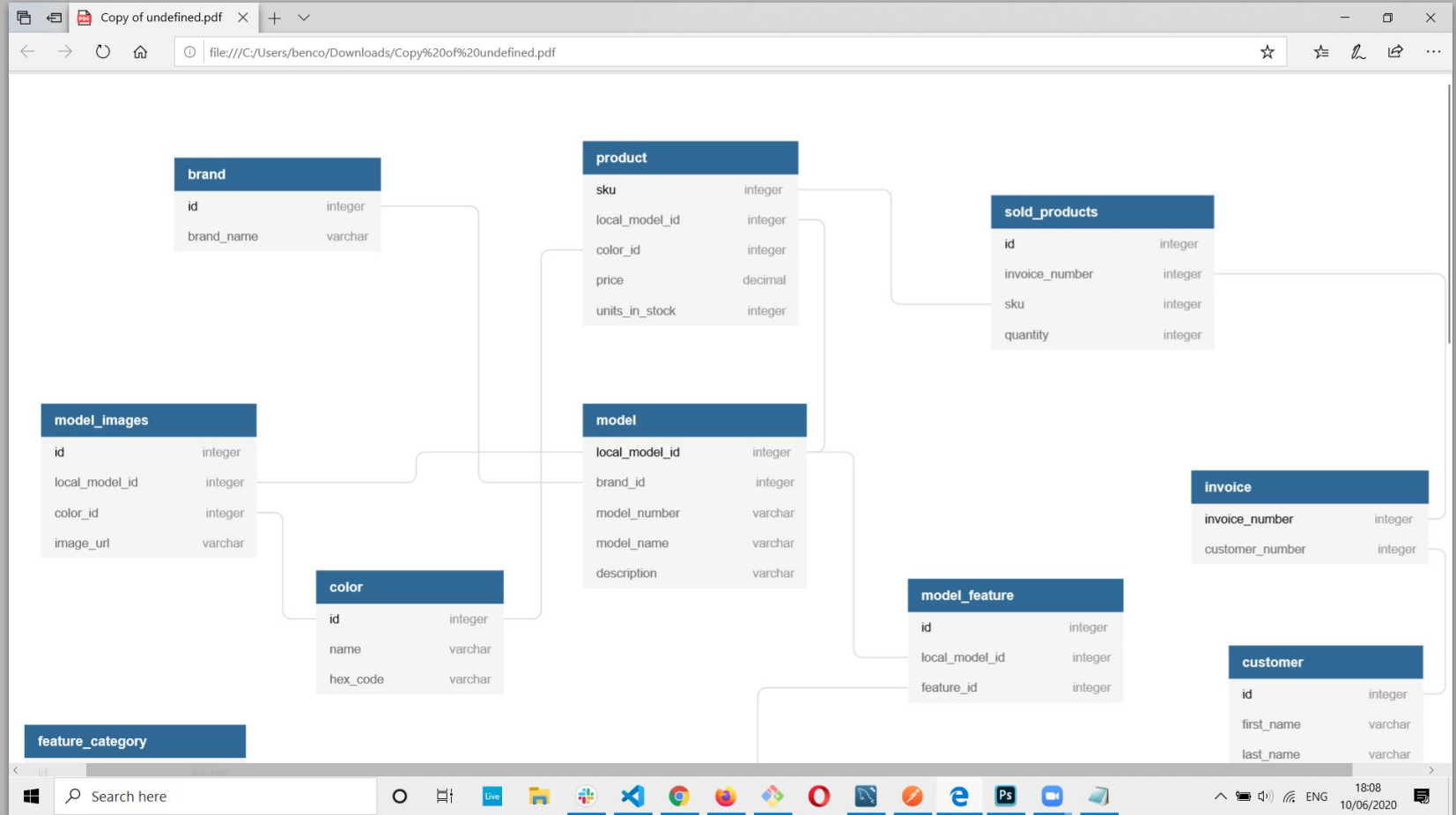
Design



GitHub best practice



GitHub best practice



Challenges

- Learning how to use sequelize to build associations in the database
- Unfamiliar with the technology
- Working virtually made the concept stage more tricky
- Aligning expectations on a short project
- New Team - 1. Forming 2. Storming 3. Norming 4. Performing
- Leaving time to consolidate code
- Uncovering obstacles earlier

Successes

- Planning
- Design
- Organisation
- Delegation
- Successful sequelize associations
- Database design
- Organised database
- Collaboration from all team members to keep master branch with relevant code

Directions for future development

- Present recently viewed items to the user
- Retargeting emails sent to users after cart abandonment
- Admin control panel for uploading new products to the database

Thank you!

