WEB601 MS1 project report by Oleg Sivers

23.08.2019

Table of contents

Git repositories

Project: https://github.com/sio2k1/WEB601PRJ

Journal: https://github.com/sio2k1/WEB601_week_1/blob/master/journal.md

Site Goals

Description

My project is about making a web application for a farming business. This will be a web site based on react.

At this web site there will be two different areas, one for public view, and one for site administrators.

As guest you can view Home page, read “about” information and look at prices and watch contacts page. Home page will be a full screen image and logo of the farm, at about page users can read some extra information about farm and its approaches to agriculture and growing animals, at this point text there is just a sample farm-related article text. Prices are basically a table with a list of prices.

As site admin, after login process user will have access to admin panel, where he can manage article text at about page and edit prices. And login menu link will change to logout.

Mission or purpose

My project is about making a web application for an existing farming business. Site will represent to audience current prices, articles about the farm. The audience is basically are clients of the farm and new clients attracted by advertisement in the internet.

Goals

Short term goals are attracting clients on site, display them an actual price, contacts information and some articles about farming. For site admins – make a tool to modify content of the web site.

Long term goal is automation of client ordering processes using a web app.

Clients

New people will come to this site mainly because of advertisement, to land on landing page. Current customers will find a reliable information about prices and get some extra information about farming. In future online orders of farm production will be available to customers.

Define the User Experience

UX for a site is a critical part, an experienced web designer and UI\UX developer will develop a user interface where usability will be as important as visual design part. UX\UI is basically not only a web app concept, it can be used in any physical products.

Audience

This site is mainly target for current customers of farming business and mature audience who want to buy top quality, natural and fresh farming products.

Scenarios

Visitor should be able to:

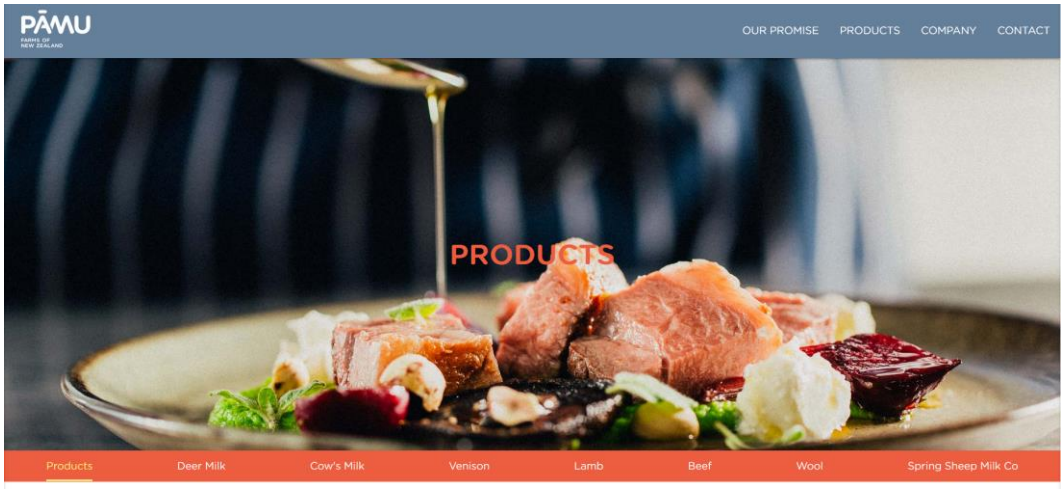
- receive actual prices on farm production
- read an information related to farm (description, way of farming, contacts)
- login (incase visitor is site admin)

Site admin should be able to modify current contents of the site.

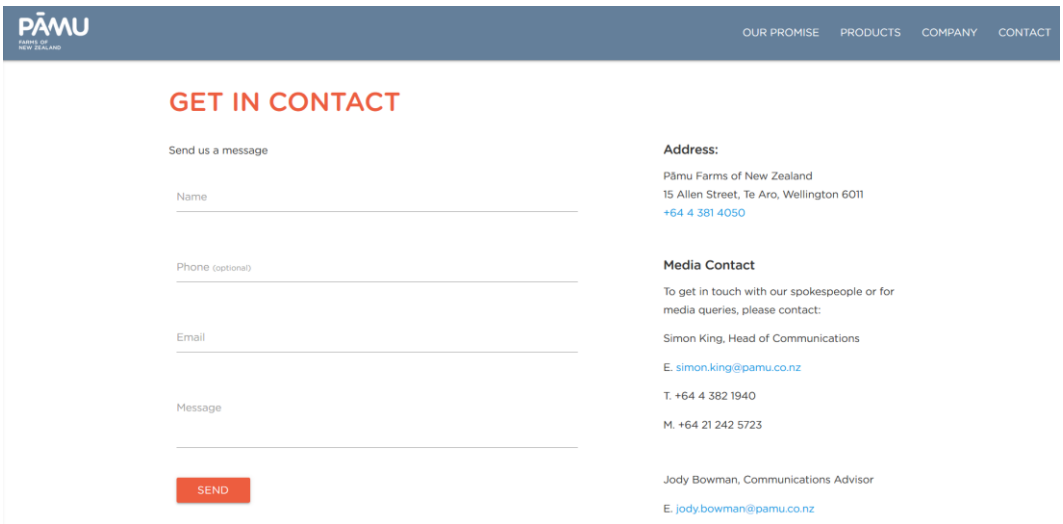
Competitive analysis

I will like to compare my site to https://pamunewzealand.com because it almost has the same idea to mine.

Its typical page:



Its contact page:



All menu pages build like SPAs, so after you click a menu link, there is a submenu available and particular page start to behave like SPA with links that scrolling pages. It looks good, but in my opinion, navigation is a little bit complicated on that site.

After looking at this site I realized that is so much work to do, in terms of design, I also liked a contact window, I'm going to add a "contact from site" functionality to my project for the next milestone.

Site Content and Structure

Metaphor exploration

Metaphors are commonly used to help users understand the new by relating it to the familiar. (Rosenfeld, Morville & Nielsen 2002) We will explain our web site functionality in simple associational way.

Organisational metaphors

My site is actually like a bulletin board, it containing some advertisement information, prices and contacts, like some person add some sale information to board, leave services of goods he or she provided and leaves the contacts. But, because it's the WWW, you can modify your announcement at board at any time.

Functional metaphors

My site will be like an newspaper's ad block, like when you look at particular advertisement in newspaper, you can read some information there, that corresponds to ad subject, you can extract some product details and get a contacts if you are interested, but in newspaper we are limited with space and text, sometimes its black and white also, while in a web site we can do some interaction, split content and make it all fancy looking.

Visual metaphors

My site representing a natural farm, so basically, I will use mild green\brown\beige color palette, because a lot of people will associate green and fresh, brown and natural. Soft colors will be used, to make site look pleasant to an eye.

Site structure listing

1. Home or "/" page, this page will be available to any visitor, you can access it from navigation menu. This page includes menu and full screen logo.
2. About, this page will be available to any visitor, you can access it from navigation menu. This page is nested from default layout page, it will include menu, article text and footer. This page will include article about farm.
3. Price, this page will be available to any visitor, you can access it from navigation menu. This page is nested from default layout page, it will include menu, price and footer. This page will include farm prices.
4. Contacts, this page will be available to any visitor, you can access it from navigation menu. This page is nested from default layout page, it will include menu, contact information and footer. This page will include farm contact information.
5. 404 page, this page will be available to any visitor, you can access it by typing a wrong route in URL. This page is nested from default layout page, it will include menu, 404 info and footer. This page will include 404-page text.
6. Admin panel, this page will be available to logged in users, you can access it from menu after login. This page will include menu, sub menu and editor to modify contents of the site. This page will include submenu for site pages and editor
7. Login page, this page will be available to any visitor, you can access it from navigation menu. This page includes menu, full screen logo and login form.

Define navigation

Global navigation

These pages will be available with persistent global routes, links will be available from navigation bar:

- Price list
- About
- Contacts
- Home page (or “/”)
- Admin
- Login

Local navigation

Local navigation will be available at the admin page, there would be editing sections corresponding to pages with content, but these sections won’t be available from application navigation menu, and possibly will not have any persistent routs.

Structure and navigation

Site will consist of several pages, which will be displayed for every visitor:

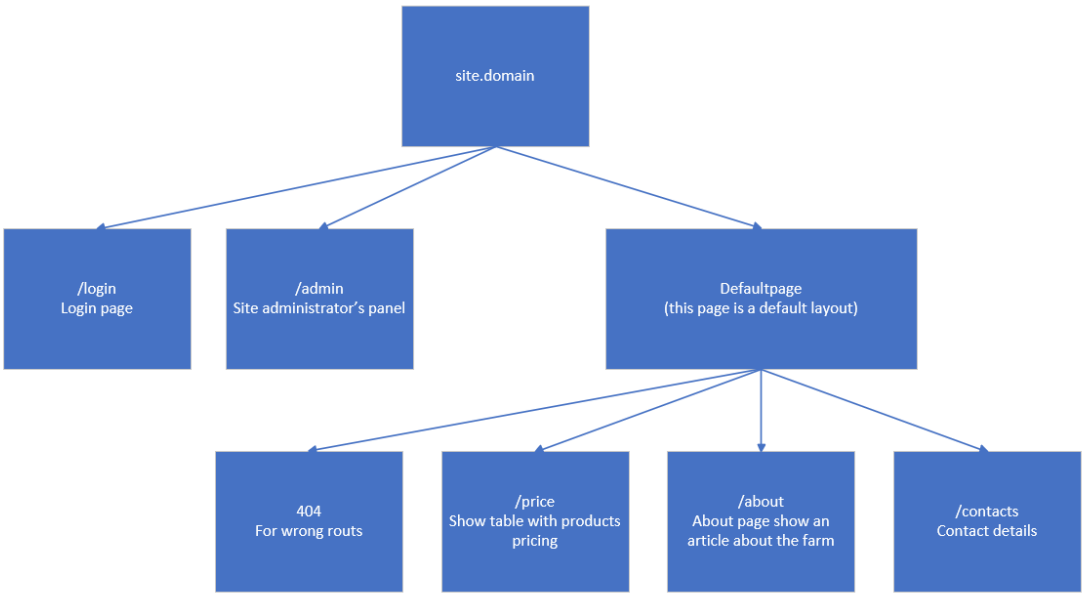
- Price list – displays price of goods
- About – article about farm
- Contacts – simple contact information
- Home page – landing page with logo
- 404 page – if route is not matching known route

Pages, that will be displayed after login:

- Admin – admin panel

Pages are divided on two groups:

- Pages, that inherit layout from default page (/price, /contacts, /about, 404)
- Page with unique layout (/home, /login, /admin)



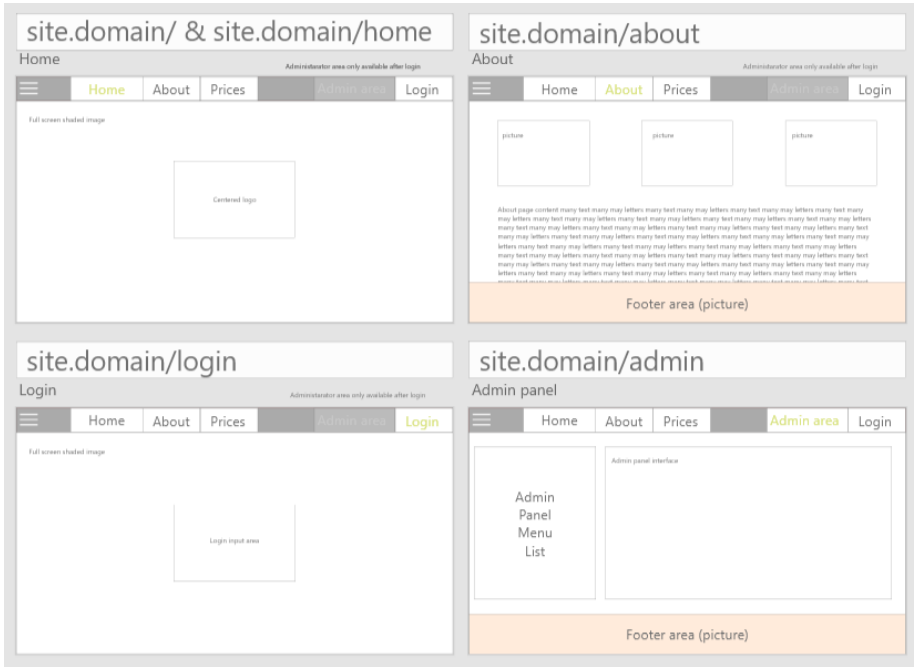
Available links:

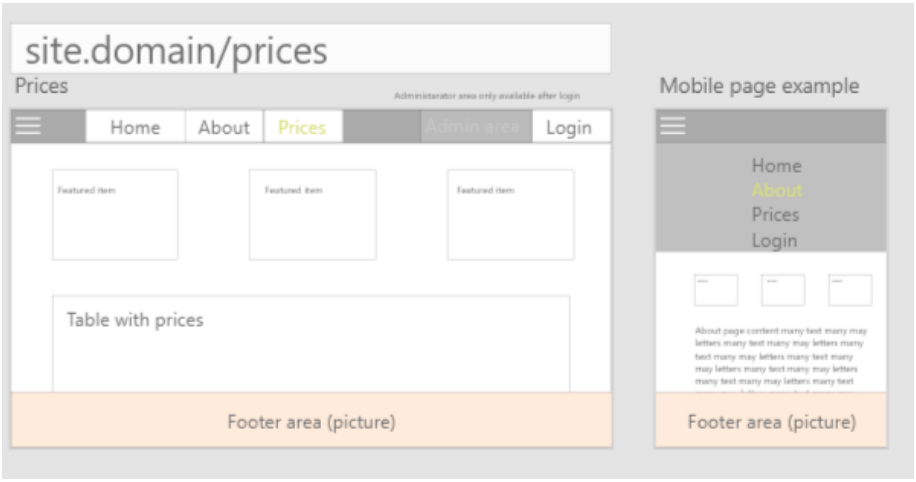
- Homepage (landing, logo): http://localhost:3000/
- About (article about farm, article text is sample text) http://localhost:3000/about
- Price (current prices) http://localhost:3000/price
- Contacts http://localhost:3000/contacts
- Login (login page) http://localhost:3000/login
- Admin page (route is not available until login finished) http://localhost:3000/admin

To login for now use credentials from json, located in /src/jsondata/users.json (I won't do this in real project, I promise!)

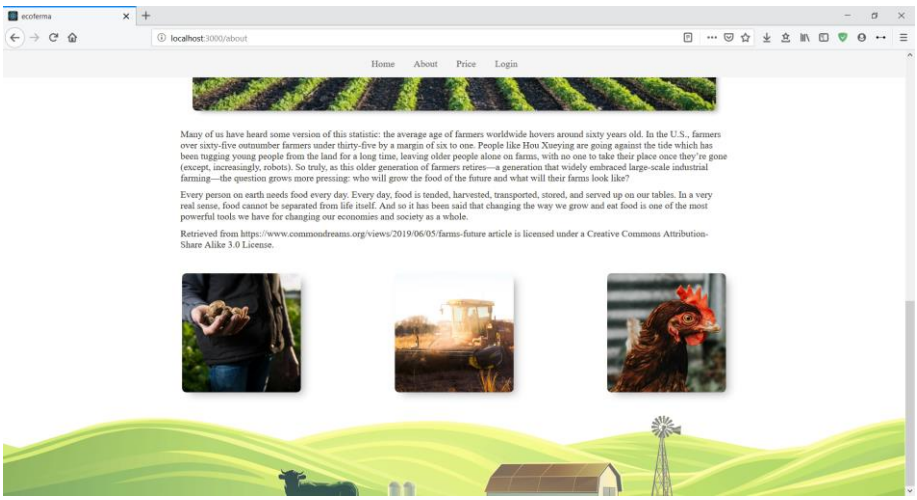
Visual Design

Wireframes are available as separate file, you can find high resolution wireframes there: https://github.com/sio2k1/WEB601PRJ/blob/master/Project_Documentation/wireframes_farm.xd





Example page:



Project tech description

For project I used an application template, generated with **create-react-app**, it actually did a lot of work for me. For creating this project I used several node.js modules, including react, react-router and redux.

Component structure

Entry point into my application is AppRoot class, defined in App.js, it rendered into in my index.html page in App.js I also create a redux store and initialize react router.

The next part of the application hierarchy is Routing class which defined in `'src/components/routing/routing'.` It's a container and it mainly defines routs of our application, 404 error page, custom pages like login and home, and pages with default layout like price and 404. This container is attached to the store for a reason, if we have a visitor, who is not logged in, we hide a route to admin page with `login_routes()` function (more information about hiding or showing parts of JSX in next paragraph). `<Switch>` component allow react-router to stop handling routes if app find matching one, that is how 404 page working, if app did not manage to find a corresponding route, then it will show 404 page.

On top of all routing there is an `App_layout` class which simply defines, that application contains menu at every single page and rest of HTML, CSS and logic is defined in {children}. Menu is coded in `Navigation` class which defined in `'src/components/menu/navigation'`, its also a container, because we need to manage our menu state according lo login state. In my application login state is defined as `state.login_reducer.user_id` if user id is -1 than we don't have any user logged in. So, we manage our menu by calling `login_menu_*` methods to append JSX. Some permanent menu items, like about are hardcoded in JSX.

`App_layout {children}` are basically our pages themselves. Pages with the same layout (content and footer), like "price" and "about" are nested from Defaultpage component, which is located at `'src/components/defaultpage/defaultpage'`. There are several pages with custom layout (for example without footer) like "home" and "login", all pages defined as a react components and located in separate folders, like "about" page is in '/components/about', "home" page is in ''/components/home' and so on.

SCSS

Project is based on SCSS as main CSS preprocessor syntax, at this point I don't use webpack. I'm using an extension for VSCODE to compile my SCSS files, called "Live Sass Compiler"

Json data

All json data is located in `'src/jsondata'` this folder is containing sample json data, in next steps of project development it will be transfered into backend. At this moment there are 3 json files:

- about_content.json (information for about page)
- prices.json (information for price page)
- users.json (logins and passwords)

To convert json into JSX I'm using parser components, one of parsers is common component, located in `'src/components/common/page_json_parser.js'`

, it converts json with paragraphs, pictures, header and footer into article. Second type of parsers is page-related parsers, for now I'm using one to parse price list json into JSX table, this one is located in `'src/components/price/json_parsers_price.js'`

Auth

Based on application state, made with redux and "Login" container. After enter credentials into form we compare credentials to match to users.json file. If there is a match we will replace our state with new state containing state.login_reducer.user_id = [id gained from json] after it, containers which are subscribed on user_id will be re-rendered according new application state. (route /admin will be available, menu will include link to admin and link to logout, link to /login will be removed from menu and /login will get redirection to /home.) Anything is related to login process located at `'src/components/login/'`

Redux

Application contains storage and several redux containers, used for controlling application state for user auth. At this point application only save its state while its opened. Redux is present in full state, I have one reducer, combined reducers, storage and actions. Storage is declared in `'src/App.js` reducers are combined at `'src/components/reducers/index.js'`. As for now I have only one reducer and one file with actions for it. They are located at `'src/components/login/redux_login_reducer.js'` and `'src/components/login/redux_login_actions.js'`

References

Rosenfeld, L., Morville, P., & Nielsen, J. (2002). *Information architecture for the world wide web*. " O'Reilly Media, Inc.".