ROAD TO HIRE
ECOMMERCE PROJECT
Phases III and IV

September 18, 2019

SUMMARY

As part of the Road to Hire program, students are engaged in the design, build and deployment of an ecommerce site. The site is constructed in stages over the course of the program to help reinforce concepts and refine skills in commercial application development.

The ecommerce site is narrowly scoped. By the end of the program, students will have completed that includes

- Home page with slider images
- Product page that allows the user to filter by product type and price
- Contact page with form
- Footer with social media links (can link to students' pages)
- Backend API server and database server
- Admin site for managing products and viewing contact requests
- Responsive pages for various device types
- Accessibility and security.

A summary of work accomplished in the first two phases follows:

Phase I: Students built static site with HTML, CSS, Bootstrap & jQuery. The site was mocked up in code

Phase II: Students created functional products page using React. Data was placed directly in state at this point since students hadn't built persistent data stores at this point.

In **Phases III and IV**, a MYSQL database will be designed and deployed on desktops to accommodate the ecommerce database. The ecommerce database will be comprised of 3 tables, storing information product, pricing, and contacts.

With the MYSQL database designed, deployed and tested, students will construct and api server using nodejs, refactor their product page from Stage II, and integrate their webpages, api server and database to meet a set of requirements.

PROJECT REQUIREMENTS

TABLE I

Req #	Title	Description
1	Database Design	An Entity Relationship Diagram describing the relationship between the three tables in your ecommerce database. The ERD diagrams will be stored in a directory of the project entitled 'design'
2	ecommerce schema	A schema for a MYSQL database defining 3 tables (product, price and contacts). This schema will be tested on MYSQL workbench, and the final schema saved in a directory in the project entitled 'dB schema'
3	Database seed	Seed data for the ecommerce database. A minimum of 10 products with prices and 10 contacts will be mocked, and seed in the database. The seed data will be stored in the 'dB schema' directory of the project, in the form of SQL INSERT scripts
4	API requirements (see details in Table II)	The API requirements are a given this phase of the project. The requirements will be captured and recorded in the 'design' directory of the project. Any modifications (customizations) made to the requirements should be reflected in an updated set of design documents in the design directory
5	API Server	Design, build and test a NodeJS server to handle api endpoints consistent with the set of specified API requirements (Table II). The server should include the following technologies, dependencies and middleware: express and express Router MySQL (client integration to database) morgan (logging) helmet (security) dotenv (password protection)

		The server should take advantage of the Error Class in nodejs for managing exceptions The API server receives requests from the Product page and responds accordingly. In some instances, the API server will need to retrieve information form the MYSQL database to complete a function
6	Product page refactoring and integration	The ReactJS product page developed in Phase II should be refactored to initiate and handle the set of API calls required in this project (Table II). The product and contact page will most likely need to be updated to handle data retrieved by the APIs as well as CRUD methods. All APIs are directed towards and resolved by the API server as defined in Requirement 5. It is recommended that you use 'fetch' in your ReactJS app
7	Automated Tests	A set of automated tests will be provided using Mocha and Chai, demonstrating the integrity of the server APIs.
8	Project Organization	The ecommerce application will be structured in a single directory, containing the client, server and other artifacts such as design docs or schema definitions. Include a README.md with appropriate docs about the app. Ince a LICENSE. Once the MYSQL server is seeded with the ecommerce database and data, the app is started using an NPM script. The npm modules 'concurrently' and 'if-env' will be needed. The project directory should be well structured and include '.gitignore' to ensure that only appropriate project artifacts are version controlled on github
9	Project demo and presentation	To be scheduled

API ENDPOINTS

TABLE II

API endpoint	Method	Description
Meets minimum requirements		
/	get	Index route to serve home page
/api/products	get	Returns a JSON object with all products and associated prices from the product and pricing tables.
/api/contacts	get	Returns a JSON object with all contacts. A webpage with the json dump can be used to render data
/api/productfilter/:query	get	Returns a JSON object with the subset of product data based on the 'query'. The query can be structured in any way needed to convey the query parameters based on design
Note: navigation in the ReactJS product page		Note that the ReactJS app that was built in Phase II is a single page app with it's own navigation. It is expected that a user will still be able to navigate to Contact Page with no server api call
Bonus points for each API completed below		
/api/productinvoice/:query	get	An api which returns a JSON with the extension of an order quantity and price for the selected product. The price and product are retrieved from the ecommerce database. The order quantity is passed from the product page. A flat tax of 8% is also added to the total invoice amount
/api/newcontact	post	An api which updates the contact table on the ecommerce database with a new contact information. The contact page should indicate that the new contact info was successfully updated. The process should include field level validations and edits
/api/deletecontact	delete	An api which deletes the selected contact from the ecommerce database, with appropriate message indicating status on the page
/api/updatecontact	put	An api to update information for the selected contact with all appropriate validations and edits