Costa Rica Institute of Technology

Academical Area of Computer Engineering

I Semester 2020

Code name: **TECBox**

Team

- **Kevin Cordero** - *Lead Developer on Back-end and Connections* - Skryfall
- **Alejandro Ibarra** - *Lead Designer. Mobile App Developer* - AlejandroIbarraC
- **Jose D. Sánchez** - *Functionality Manager & Web Developer* - JoseDavidSS
- **Jesús Yamir Sandoval** - *Project Manager & Web Developer* - shuzz22260427

Technical Advisor:

- **Marco Rivera Meneses** - *Teacher*

Course:

- **Databases** - CE 3101

Languages used:

- C#
- JavaScript / TypeScript
- Swift
- MEAN stack tools (Node, Angular, Express)

A. Developed structures.

For the development of the program, a series of data structures were used, which were intended to facilitate both logical and graphical data management.

- Linked Lists:

Information management was the fundamental pillar of this project. The use of lists was recurrent and essential in the development of the above, having been used mainly in two main aspects:

1. JSON data encoding:
   Due to the need to maintain an exchange of information between the application and its respective database, the use of JSON encoding for objects (being basically a list of lists in a defined format) was imperative for communication with the server.

2. Tables.
   The second great aspect for which the implementation of lists was necessary was to be able to facilitate the handling of data when showing information to the client in graphical tables to be modified. Once the object was decoded by the client, it was treated as a list of lists, grouping information together in an orderly fashion.

B. Description of used algorithms.

For the correct development of the program, various algorithms and functions were necessary to fulfill the desired roles in every part of the code. In this section, we will proceed to describe in detail the algorithms that were used in the creation of different parts of the project.

- String to Int date algorithm.

This algorithm was used to transform a date (“2000-12-12” for example), which is a String, into the same date but using a list of integers instead ([2000-12-12]).

This works by looping through the string and transforming the numbers to integers and stops when a different character is reached, adds the date to the list and continues with the next date until the end.

- Add, remove, modify algorithm.

This one was used for information management and it works like this: If I want to insert something, I add it to the end of the list. If I'm going to modify something, I look for it by means of an identifier that is usually its primary key and I overwrite it. Finally, when I delete something, I look for it the same way, but instead of overwriting it, the item is removed from the list. After the process is finished, a new JSON object is encoded to be sent to the server for storage.

- Algorithm for packages.

Finally, the algorithm to see which packages are ready to be delivered by route, which receives a number of a route and is responsible for verifying which packages are ready to be delivered on the chosen route, those who meet the criteria are inserted into a list. It works by sequentially checking if the route attribute matches the desired route.

C. Open issues

The only problem that was found involved the external library Crystal Reports. A report generator was partially implemented, what this means is: it works on its own and the reports are generated in conjunction with the database, but we couldn't configure it to work seamlessly with the project. Currently, it stands like an external executable to be run. Due to this, certain problems are generated that were not solved, and we replaced the export function in the delivery console with a table that shows the selected report.

D. Task Division

Name: *Kevin,* ***Role:*** *Lead Developer on Back-end and Connections,* ***Tasks:*** *3*			
Task	**Hours**	**From**	**Until**
API Development	11 hours	6-3-20	10-3-20
Add, Remove, Modify server logic	8 hours	10-3-20	13-3-20
Front end and back-end Connection	15 hours	13-3-20	16-3-20
Total: 34 hours			

Name: Jesus, Role: Project Manager & Web Developer, Tasks: 5			
Task	**Hours**	**From**	**Until**
Daily log	2 hours	6-3-20	17-3-20
Administrator view organization	3 hours	7-3-20	7-3-20
Creation of dynamic tables to show information	18 hours	7-3-20	11-3-20
REST API methods from front-end to back-end	10 hours	11-3-20	15-3-20
Documentation	4 hours	15-03-20	16-03-20
Total: 37 hours			

Name: Alejandro, Role: Lead Designer. Mobile App Developer, Tasks: 3			
Task	Hours	From	Until
Project Design and Web Development	18 hours	10-03-20	14-03-20
Mobile App Development	11 hours	06-03-20	10-03-20
User Manual & Additional Documentation	6 hours	14-03-20	16-03-20
Total: 35 hours			

Name: José, Role: Functionality Manager & Web Developer, Tasks: 3			
Task	Hours	From	Until
Product Cart Functionality	9 hours	06-03-20	10-03-20
Partial Internal Documentation	7 hours	11-03-20	13-03-20
Login View	3 hours	14-03-20	16-03-20
Total: 19 hours			

E. Closed issues

- **Table problems.**

There was initially a recurring problem with the tables that were used to display the information. This problem threw us an error when compiling the project as such. The root of the issue was a small syntax error with the TypeScript file in which the information to be displayed in the tables was handled.

As a tip it is recommended to be very careful with the methods implemented for data management and its implementation syntax.

Useful help:

Pipes in angular: https://angular.io/guide/pipes

How to build a table in angular with Pipes: https://stackoverflow.com/questions/47631908/filter-on-multiple-columns-in-a-table-using-one-pipe-in-angular

- **Problems with POST and GET methods to server.**

Another big problem we had during the project occurred when transmitting information between back-end and front-end.

For this specific part it was decided to use **axios** (a HTTP request library), because it was investigated beforehand that it was quite useful. However, there were certain problems when translating the syntax of POST methods and receiving information from the API. As a tip, it is recommended to read the documentation well when setting up the communications between front-end and back-end to ensure a seamless experience.

Useful help:

Axios tutorial: http://zetcode.com/javascript/axios/

HTTP requests with Axios: https://styde.net/solicitudes-http-con-axios/

F. Conclusions and Recommendations

- Conclusions.

-The MEAN Stack is one of the most powerful and useful tools for web development

-Aesthetics is of utmost importance for creating a good website.

-Diagrams are of vital importance when creating a database.

-Tools like NodeJS and Bootstrap are essential for any web developer to knowg. These can really make development easier.

-Being organized is extremely important when creating a database, since the slightest error can cause great long-term damage to its implementation or use.

- Recommendations.

-Read the Angular documentation before even thinking of starting the project. If not familiar with HTML, CSS and JavaScript code, learn that syntax first.

-Make a good and clear diagram before starting development of any database.

-Investigate as much as possible the correct and simple way to request data using HTTP from a RESTful API.

G. Development Blog

06-03-20:

The first days of the project were used for superficial investigations about the framework that was going to be used for development, as well as the installation of the necessary tools to start.

The assignment of roles and division of tasks began taking into account the weak and strong points of each member of the group, this in order to have a comprehensive and complete project. Likewise, a series of meetings were held virtually using the Discord portal. These meetings and the subsequent ones were to be held through the platform channel called "Régimen Corderista", which was established by the 4 members of the group together.

The meetings ended and each member of the group obtaining their role assignment, which is why the project started remotely and with each member fully engaged in its strong part.

Main sites consulted this day:

https://angular.io/docs

https://getbootstrap.com/docs/4.1/getting-started/introduction/

https://docs.npmjs.com/cli/install

07-03-20 to 10-03-20:

The following days were used to start making the first designs, both aesthetic and diagrammatic. In addition to this, the GitHub repository and the various views for the websites began to be created. The folders and views with which they were going to work were established, said views subject to future changes, these being only first impressions and sketches made as the skeleton of the project.

The creation of dynamic tables and database format were also investigated, to get an accurate feel of what we wanted to portray and how we wanted to do it. It was decided to present the information by means of tables, these were to be generated dynamically and with support for objects, in order to maximize the local and remote compatibility of the information that could be managed.

During the work meetings through the "Régimen Corderista" channel, it was also established, as previously stated, the method with which the JSON encoded objects that made up the project database were to be managed.

Main sites consulted these days:

https://www.figma.com/file/SeBzob8LqqFSdYeuil2cvv/Client-View?node-id=0%3A1

https://angular.io/cli/build

https://app.lucidchart.com/documents#/dashboard

https://stackoverflow.com/questions/47631908/filter-on-multiple-columns-in-a-table-using-one-pipe-in-angular

13-03-20 to 14-03-20:

These two dates were of utmost importance for the project because a meeting was held that included every member of the group. The optimal method to design the internal and external elements was discussed. Here, the main structures for encoding data internally through JSON were established, as well as very useful tools like Canva and Figma to design a good looking web app.

Another extremely important point that was discussed at this time was the handling of information between the backend and the frontend. It was concluded

that the optimal method to achieve this objective was to use the Axios external library for easy and elegant HTTP requests.

In two of these meetings, the colleague and member of the group assigned to the role of managing the project and much of the web development, could not attend the meetings due to animic and academic problems, for which the members of the group Alejandro, Kevin and Jose temporarily covered him in his activities.

The mobile app was also quickly developed in this time. The syntax and feature set of the framework we chose for this part, SwiftUI, was rapidly understood, as we had already worked with the Swift programming language for the development of mobile apps in the past. Also, the overall aesthetic design language and styles were set in stone for every view to adapt.

Main sites consulted these dates:

https://www.crystalreports.com

http://zetcode.com/javascript/axios/

https://blog.hotmart.com/es/canva-tutorial/

https://www.hackingwithswift.com

16-03-20 to 17-03-20:

The last stage of connecting everything together was planned for this week. However, due to the COVID-19 global pandemic, the team did not have a clear idea of when the project was due. During this time, the table functionality was implemented, as well as the final definition of the database and RESTful API. Also, the overall look and feel of the web app was defined, along with the operating mechanics of user access, authentication and cart product management.

Main sites consulted these dates:

Creating Shopping Cart Using Angular8

https://medium.com/@moshevilner/creating-shopping-cart-using-angular8-c6a5d4a0f0b3

The same sites in previous post of this development blog.

27-04-20 to 20-05-20:

In the final stages of the project everything flowed almost seamlessly, being the only problem found the one that was announced by Kevin, main backend manager. In this report he exposed the difficulties encountered during the management of the Crystal Reports library, bringing this to the members of the group to be forced to add this as the only problem, although without affecting general functionality of the project, which could not be solved.

The user manuals and the installation manual were completed, in both cases we used Canva as a tool to format the document. Likewise, the final class diagram and the comparison between initial and final conceptual database diagrams for the project in general were made.

The database was successfully completed, and the final touches were given to the CRUD's functionality in Axios.

Also, a whole lot of bugs were fixed, especially with how some things looked and worked. The mobile app was finally connected to the API using a ridiculously easy-to-use library for Swift HTTP requests named Alamofire. At this stage of the development, we had enough time on our hands and everyone took their time with every task. It was a nice change from our usual intense workflow.

The last meeting held by the "Régimen Corderista" channel was to touch on the final points of the project, perform functionality tests and final touches, concluding with a successfully completed project. With all the members having finished their tasks, the external documentation was carried out jointly in order to present the important points made in the project. Now I conclude magnanimously, after many hours of work, the development of such an arduous task, a never-ending project that has finally reached its end.

Main sites consulted these dates:

https://app.lucidchart.com/

https://www.canva.com

https://bezkoder.com/react-hooks-crud-axios-api/

H. Links

GitHub repository: https://github.com/AlejandroIbarraC/TECBox

Figma app mockup design:
https://www.figma.com/file/SeBzob8LqqFSdYeuil2cvv/Client-View

Canva TECBox User Guide:
https://www.canva.com/design/DAD8WJVbZD4/zAD0i00RrYII9nUzS0L6Pg/view?utm_content=DAD8WJVbZD4&utm_campaign=designshare&utm_medium=link&utm_source=homepage_design_menu

Canva TECBox Deployment Guide:

https://www.canva.com/design/DAD8w9N6VNU/o_GYOSV-Xf12nC8iD_f3dg/view?utm_content=DAD8w9N6VNU&utm_campaign=designshare&utm_medium=link&utm_source=homepage_design_menu

H. Summarized Bibliography

Angular Documentation. 2020. https://angular.io/docs

Alamofire 5 https://github.com/Alamofire/Alamofire

Axios tutorial http://zetcode.com/javascript/axios/

Bootstrap Introduction. 2020. https://getbootstrap.com/docs/4.1/getting-started/introduction/

Build using Angular framework 2020. https://angular.io/cli/build

Canva. ¡domina esta herramienta! https://blog.hotmart.com/es/canva-tutorial/

Creating Shopping Cart Using Angular8

https://medium.com/@moshevilner/creating-shopping-cart-using-angular8-c6a5d4a0f0b3

Crystal Reports. https://www.crystalreports.com

Filter on multiple columns in a table using one pipe in Angular. 2018

https://stackoverflow.com/questions/47631908/filter-on-multiple-columns-in-a-table-using-one-pipe-in-angular

Hacking with Swift https://www.hackingwithswift.com

How to install with NPM https://docs.npmjs.com/cli/install

Lucidchart https://app.lucidchart.com/documents#/dashboard

React CRUDS Hook Example with API https://bezkoder.com/react-hooks-crud-axios-api/