Decoupling the UI Framework from Business Logic for Fusion Corporate Channel

Anusha[1], Renjith Manhachery[2]

[1]*Student, Electronics and Communication Engineering, RVCE Bengaluru-59.*
[2]*Director Development, Fusion Corporate Channel, Finastra Software solutions, Bengaluru-37.*

---***---

Abstract - *With the technological advancement in the 21st-century, everybody wants to experience the best technological capabilities and facilities with least spending. User experience consists of how a person feels during a machine interaction. It involves all facets of the relationship between the end-user and the business, its services and products. In this project we have developed an interactive web application with authenticated access, displaying an emulation of the existing product. Further, developed an user friendly Interface for the Fusion Corporate Channel with improved quality and evaluate the functionality of the developed software application to ensure that it has met the specified requirements and is defect free. In this paper We have discussed the steps involved in implementing the project, also the before and after stage of the improvement is shown in the result along with the performance analysis.*

***Key Words*: Fusion Corporate Channel, user interface, REST API, primeNG, Web application.**

1. INTRODUCTION

User experience (UX) design is the process of designing products that are useful, simple to use and time-saving. This consists of optimizing the overall experience of users engaging with a company and ensuring they get interest, happiness and enjoyment. After understanding and learning the required web development skills, the web application is developed with Visual Studio Code editor using angular framework. Firebase application is used to implement authentication and database management. Further quality assurance and code testing is done with the assistance of JIRA tool for following the status of workflow and Github to keep track of the code changes. The remaining part of the paper has been written as follows, section 2 discusses the related works carried out in the field of web development and user interface. Section 3 discusses the design and implementation of methodology for developing the web application. Section 4 discusses the improvisations done on the existing FCC product. Section 5 discusses the results obtained.

2. RELATED WORKS

In [1], The authors address REST API Management and Development Using Model Based Architecture. REST (Representational State Transfer) which is the most commonly used means of building, publishing and accessing Web Resources, using JSON (JavaScript Object Notation) as a medium for data sharing. Nevertheless, REST (like WSDL for Web Services) and JSON (like XML Schema for the XML language) are not official standards. The authors focus in this paper on managing and creating REST resources based on the MDA (Model-Driven Architecture) concept which enables complex project design and maintenance. It provides a way to define the REST API in MDA and to provide automated evolution control between subsequent versions of the API that are derived from the original. The approach suggested describes a novel model that reflects REST requests and algorithms to provide evolution of this model based on improvements made in MDA's PIM (Platform-Independent Model).

The paper [2] proposes an AngularJS UML profile to construct an AngularJS web application model and turn the model into a template code. In view of the program specifications, a device analysis vendor uses a UML modeling tool to construct an implementation model based on the AngularJS UML profile (step 1). The code is first converted into the M2C (Code-to-Code), which is the development method that understands the development models (stage 3), which is then translated into the XMI format (stage 2). The consequence is a source code prototype for AngularJS (step 4).

3. DESIGN AND IMPLEMENTATION

The flow of the project will be as shown in Fig 1. The initial stage of the project is spent in learning the skills required for web development followed by developing a web application with the specified requirements. In the FCC UI, a thorough testing is done to detect any inadequacy in the functionality, any issue detected is analyzed and an appropriate solution is obtained, this step is repeated until the application meets all the required standards.

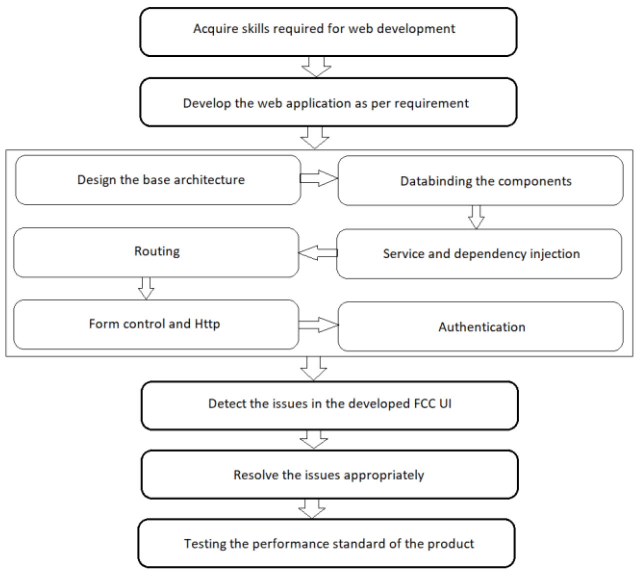

Acquire skills required for web development
Develop the web application as per requirement
Design the base architecture
Databinding the components
Routing
Service and dependency injection
Form control and Http
Authentication
Detect the issues in the developed FCC UI
Resolve the issues appropriately
Testing the performance standard of the product

Fig -1: Methodology of the project

The FCC provides several services to the customers like Account services, Cash services, Trade service, Loan etc. The base architecture of the emulated application can be represented as shown in Fig 2. The header consists of two tabs-dashboard and Event list. The dashboard consists of the list of available events which provides their details on selecting the event and the provision to add the event items to an individual's account. Similarly, the Event list consists of an individual's list of events.

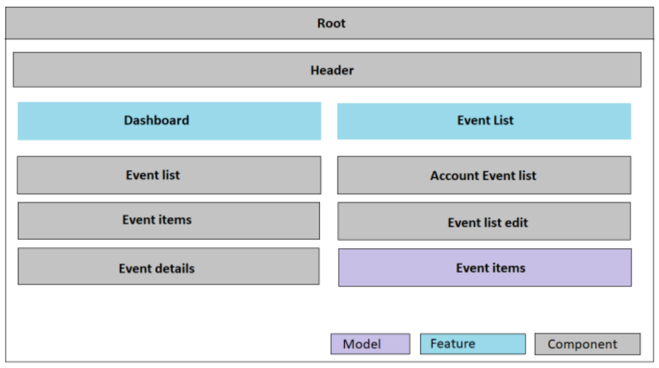


Fig -2: Application Architecture

Model is Maintaining Data models i.e. it is a service which will pass data to the components based on request. Feature refers to the attributes in the web page. The components are used to split to application into smaller parts, they are used to create UI widgets.

The detailed flow of application is as follows-1) Designing the base architecture: The base page building starts with the navigation bar consisting of required tabs and components which control a patch of screen called a view. We also create the required models which will pass data to the components based on request. 2)Databinding with the components: Data binding is used to perform automatic synchronization of data between the model and view components. We also build dropdown directives and use click listeners to add the events to the list. 3)Services and dependency injection: Service is used to integrate and exchange business principles, templates, or data and functions with various Angular device components. We employ service to push the selected event to the individuals Account list. 4)Routing: An angular router will use the URL of a website to connect to a client page. The supporting view component can pass optional parameters to help it decide which particular content will be displayed. We link the router to the links on the page and navigate to the appropriate view of the application when the user clicks the link. When the user taps, selects from the drop box or in answer to any other trigger from any source, we will navigate imperatively. 5)Form Control: Every form input present in a reactive form should be bound by a form control. All the required conditions like form validation, updating and deletion can is managed through form control. 6)Http: The HttpClient in the http service of the angular/common module, offers a streamlined Web client API for Angular applications that rests on the browser-exposed XMLHttpRequest interface. We make use of this feature to integrate Firebase application[3] in our project which is used to implement backend and authentication. 7)Authentication: All the authentication actions are implemented and integrated with the application.

4. IMPROVISATION OF THE PRODUCT

After the initial development phase of the application, the product is tested for quality and compliance with the standard guidelines. The original codes of FCC product are stored in bitbucket repository, which is extracted based on requirement. The extracted code is modified using the Visual studio code[4] and the changes are kept track using Github[5]. Eclipse application is used to connect the Oracle database[6] with the server and launch the server[7]. The management of the workflow is done with the assistance of JIRA tool[8] . This section discusses some of the issues that were detected and how it was resolved to improve the quality of the product.

For every Angular component, We can describe the CSS styles that are used in the sample, defining any selectors, rules, and media queries that are required. This can be done by setting the styles property in the component metadata.

4.1 PrimeNG Dropdown

Dropdown needs a binding value and a set of options. There are two alternatives for specifying the option property; one way is to provide a set of SelectItem instances, while the other way is to include an array of random objects along with the optionLabel property to define the field name of the option. The SelectItem API is designed to provide more control over how the options are presented, such as grouping and disabling. In the project we make use of Custom dropdown wherein both the selected option and the options list can be templated to provide customization on the default behavior which is displaying label property of an option. We use "selectedItem" template to customize the selected label display and the "item" template to change the content of the options in the dropdown panel. In addition, zwhen grouping is enabled, "group" template is available to customize the option groups. All templates get the option instance as the default local template variable. If the options are custom objects not SelectItem instances, we can use value property to access the custom object.

4.2 JIRA integration with Bitbucket Repository

Bitbucket provides us a provision of integration with Jira. Bitbucket and Jira Cloud are independent services, we can have both a Jira Cloud account and a Bitbucket squad, each with its own collection of accounts, permissions and access laws[9]. Bitbucket teams are not accounts i.e. they must be handled by an administrator (or administrators) who has separate Bitbucket accounts. Every member of the Bitbucket team must have their own individual Bitbucket account. When new team members are invited using their email they are also automatically invited to sign up for Bitbucket and are automatically added to the team when they complete sign up. Bitbucket repository ownership can be transferred to a team: this can be helpful if you want to create a team based upon existing repositories. Following are the advantageous features of integrating Bitbucket with Jira 1) Automatically connection problems and keep the staff up to date. Build a branch from a Jira issue, or connect the issue keys to a commit, branch, or pull the request to link it. 2) Get a working environment and renew the team without swapping applications. Demonstrate, edit, update, and more about Jira tickets in the Bitbucket Cloud UI. 3) Assess the success of the Jira problem at a glance. Display and develop branches, pull requests and view commits without leaving Jira. The codes relating to our concerned issue are extracted from the repository and are merged back after making the required amendments and getting the approval from concerned people.

5. RESULTS AND DISCUSSION

In this section, we discuss about the developed application and its feature, followed by the results obtained after resolving the issues raised in the FCC product. We also discuss the parameter analysis which shows that the techniques we have used is better compared to the existing ones.

Outcome of the developed web application

The application developed contains features for both the administrator and the customer. Admin operations include adding a new event in the homepage and making the required modifications whereas the customer user have the provision of managing the event actions. They can own and manage personalized event list. On opening the application, it opens an authentication page for login. There are two possible scenarios in this situation, the user can either sign up as a new user or login if already registered. Following are the considerations that are taken care of-1)A new user is allowed to sign up. 2)An existing user can login with right combination of username and password. 3)An error message "User already exist" is displayed when an existing user tries to sign up. 4)An error message "Password is incorrect" is displayed when user enters an invalid password. We also see that the login button is disabled unless valid entries are provided in the required field, shown in Fig 3. The error messages displayed are shown in Fig 4.

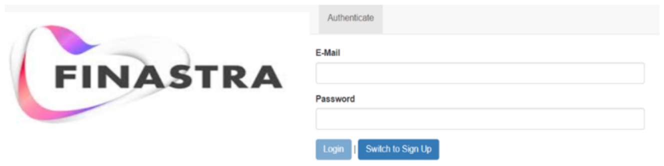
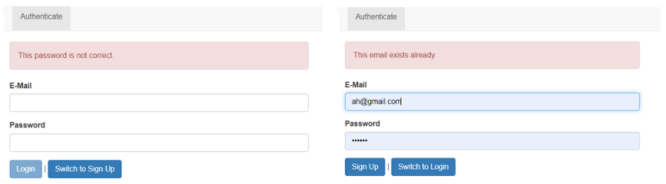


Fig -3: The login screen



This password is not correct

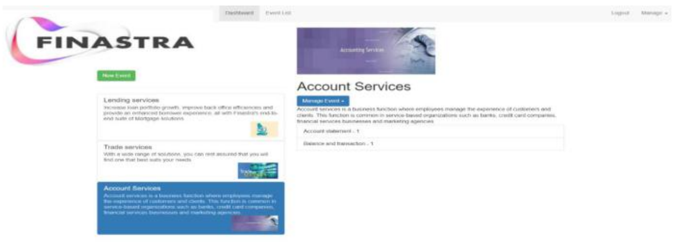
This email exists already

ah@gmail.com

••••••

Fig -4: Login screen with error messages

The home page appearing on the screen on logging in is shown in Fig 5. It displays the list of existing events like Lending service, Trade service and Account services. Each event has provisions of multiple actions like Account services has account summary, balance and transaction etc.



Lending services

Increase loan portfolio growth, improve back office efficiencies and provide an enhanced borrower experience, all with Finastra's end-to-end suite of Mortgage solutions.

Trade services

With a wide range of solutions, you can rest assured that you will find one that best suits your needs.

Account Services

Account services is a business function where employees manage the experience of customers and clients. This function is common in service-based organizations such as banks, credit card companies, financial services businesses and marketing agencies.

Account Services

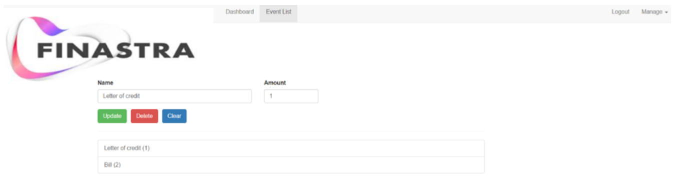
Account services is a business function where employees manage the experience of customers and clients. This function is common in service-based organizations such as banks, credit card companies, financial services businesses and marketing agencies

Account statement - 1

Balance and transaction - 1

Fig -5: Home page of the application

The manage dropdown at the top right corner of the page is used to save and fetch the data from the database. The user can select the actions required for their use and add it to their own list as shown in Fig 6.



Name: Letter of credit

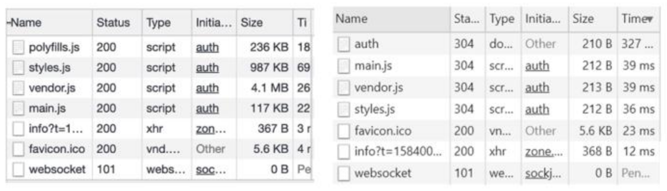
Amount: 1

Letter of credit (1)

Bill (2)

Fig -6: Event list of the user

The implementation of dynamic function loading(DFL) is used for the selective loading of pages which considerably reduces the resource size. A comparison of the resource size with and without the implementation of dynamic loading can be seen in Fig 7.



Name	Status	Type	Initia...	Size	Ti
polyfills.js	200	script	auth	236 KB	18
styles.js	200	script	auth	987 KB	69
vendor.js	200	script	auth	4.1 MB	26
main.js	200	script	auth	117 KB	22
info?t=1...	200	xhr	zon...	367 B	3 r
favicon.ico	200	vnd....	Other	5.6 KB	4 r
websocket	101	webs...	soc...	0 B	Pe

Resource size before dynamic function loading

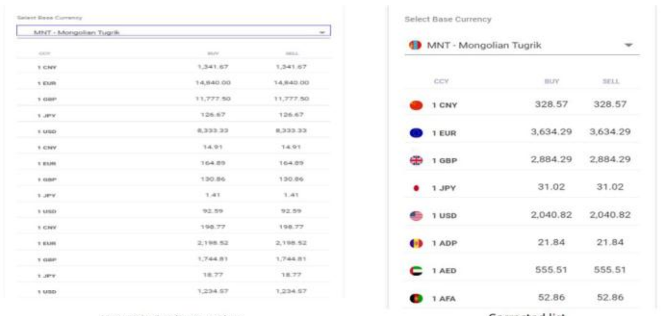
Name	Sta...	Type	Initia...	Size	Time
auth	304	do...	Other	210 B	327 ...
main.js	304	scr...	auth	212 B	39 ms
vendor.js	304	scr...	auth	213 B	39 ms
styles.js	304	scr...	auth	212 B	36 ms
favicon.ico	200	vn...	Other	5.6 KB	23 ms
info?t=158400...	200	xhr	zone....	368 B	12 ms
websocket	101	we...	sockj...	0 B	Pen...

Resource size after dynamic function loading

Fig -7: Resource size before and after the implementation of DFL.

Results of the resolved issues
We had discussed some of the issues detected in the FCC product which needed improvisation. Following are the discussion and implementation of resolving issues:

1)Duplicate Listing of values in currency exchange rates widget: On analyzing the problem, it was realized that this was happening due to fast changing of currency through the keyboard, because of which the array in which data gets stored does not gets cleared at the same rate and hence multiple values are displayed. To overcome this shortcoming, we checked the JavaScript events and selectively applied the JavaScript Events: click <button onclick="hello()">Click me</button>;Double Click <button ondbclick="hello()">Click me</button>; Right Click <button oncontextmenu="hello()">Click me</button>; Mouse Hover <onmouseenter>; Mouse Out <on mouseout>; Key UP<on mouseup>; Scroll<on scroll>; Load<on load>; Unload<on unload>.The before and after state of the situation is shown in Fig 8.



Select Base Currency

MNT - Mongolian Tugrik

CCY	BUY	SELL
1 CNY	1,341.67	1,341.67
1 EUR	14,840.00	14,840.00
1 GBP	11,777.50	11,777.50
1 JPY	126.67	126.67
1 USD	8,333.33	8,333.33
1 CNY	14.91	14.91
1 EUR	164.89	164.89
1 GBP	130.86	130.86
1 JPY	1.41	1.41
1 USD	92.59	92.59
1 CNY	198.77	198.77
1 EUR	2,198.52	2,198.52
1 GBP	1,744.81	1,744.81
1 JPY	18.77	18.77
1 USD	1,234.57	1,234.57

List with duplicate value

Select Base Currency

MNT - Mongolian Tugrik

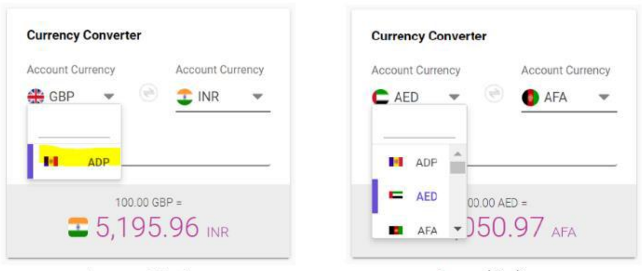
CCY	BUY	SELL
1 CNY	328.57	328.57
1 EUR	3,634.29	3,634.29
1 GBP	2,884.29	2,884.29
1 JPY	31.02	31.02
1 USD	2,040.82	2,040.82
1 ADP	21.84	21.84
1 AED	555.51	555.51
1 AFA	52.86	52.86

Corrected list

Fig -8: Currency exchange list before and after resolving

2)Hand symbol displaying outside the button: The cursor property determines the mouse cursor to be displayed while moving over an item. On analyzing the code, it was understood that there was a conflict between the styling associated with this particular element and the common styling properties of the entire component. This problem was overcome by making the necessary changes in the code.

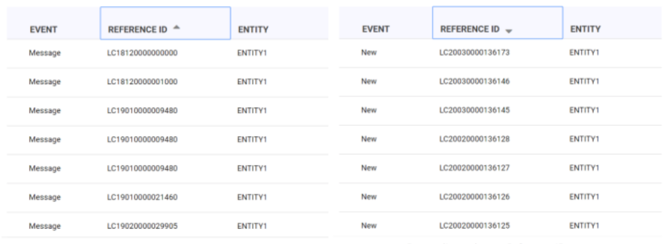
3) Improper selection in currency converter: On analyzing the problem it was understood that it was happening due to the improper binding of the label and its corresponding values. To overcome this shortcoming, a thorough study of primeNG dropdown template was done and the required corrections were made in the codes. Fig 9 shows the defect present in the product and the same after resolving.



Incorrect binding Proper binding

Fig -9: Currency converter before and after resolving

4)Enable sorting of table values: Sorting the values of the table in a particular order based on the requirement is an important feature. It was found that this option was missing in a particular table. To overcome this shortcoming, we use an API reference for Angular Material sort named MatSortModule. MatSort and mat-sort-header are used to add sorting status and display tabular data, respectively. Sorting enabled table is shown in Fig 10.



EVENT	REFERENCE ID ▲	ENTITY
Message	LC18120000000000	ENTITY1
Message	LC18120000001000	ENTITY1
Message	LC19010000009480	ENTITY1
Message	LC19010000009480	ENTITY1
Message	LC19010000009480	ENTITY1
Message	LC19010000021460	ENTITY1
Message	LC19020000029905	ENTITY1

Ascending order wrt Reference ID

EVENT	REFERENCE ID ▼	ENTITY
New	LC20030000136173	ENTITY1
New	LC20030000136146	ENTITY1
New	LC20030000136145	ENTITY1
New	LC20020000136128	ENTITY1
New	LC20020000136127	ENTITY1
New	LC20020000136126	ENTITY1
New	LC20020000136125	ENTITY1

Descending order wrt Reference ID

Fig -10: Table with updated sort icon

6. CONCLUSIONS AND FUTURE SCOPE

In this project, we have developed an interactive web application with authenticated access which emulates an existing FCC product. Further we worked on developing the Finastra FCC product and improving the user interface. We also test the interface to check the quality and ensure that the performance meets the set guidelines, and necessary changes were made and implemented when any defect was recognized to overcome the shortcoming and improve the performance quality of the product. Angular platform was used to design and develop the emulated web application along with the usage of Firebase application for backend services like data storage and authentication. For working on the existing Finastra FCC product, we extract the necessary code from bitbucket repository onto our device, manipulate the codes via Angular application, Github is used to keep track of the changes made. The workflow of any defect is tracked through the Jira platform. The issue is further tested by the testing team who check the performance, once the work is approved by the testing team the code is merged back to the repository. These steps are followed for all the defects detected or any improvisations that are required in the performance of the product. In the project we have achieved to develop an authenticated login page which allows the authenticated user to enter the application or display a relevant error message when the required conditions are not satisfied. Once inside the application, the user can perform both admin operations and customer operations. This helps in better understanding the working of the product. We also observe that implementing dynamic function loading reduces the resource size from kilobytes to bytes in this application meaning it will have a significant difference in larger applications. Further we have optimized the performance of the FCC portal by resolving some issues like duplication of values in currency exchange list was removed, the improper selection of dropdown options was corrected, the incorrect cursor action was corrected using the stylings and additional features like sorting option was included in the table, thereby improving the overall performance of the product.

The scope of the project is perpetual as the worlds market is becoming more customer led with demands to extend online banking to include new business areas and adopting many new technologies, the constantly changing customer requirements and demand opens a lot of area of research and work in this field. The existing product does not have the feature of audio web accessibility which could be incorporated in the future along with improvements in making the user interface more interactive to provide a better user experience.

REFERENCES

[1] M. Pol_ak and I. Holubov_a, “Rest api management and evolution using mda” in Proceedings of the Eighth International C* Conference on Computer Science & Software Engineering, 2017, pp. 102-109.

[2] W. Chansuwath and T. Senivongse, “A model-driven development of web applications using angularjs framework” in 2016 IEEE/ACIS 15th International Conference on Computer and Information Science (ICIS), IEEE, 2016, pp. 1-6.

[3] S. Sarkar, S. Rahman, and M. Biswas, “Secureit using firebase, google map and node. js” 2020.

[4] J. Conallen, “Modeling web application architectures with uml” Communications of the ACM, vol. 42, no. 10, pp. 63-70, 1999N. Dragoni, S. Giallorenzo, A. L. Lafuente, M. Mazzara, F. Montesi, R. Musta_n, and L. Sa_na, “Microservices: Yesterday, today, and tomorrow” in Present and ulterior software engineering, Springer, 2017, pp. 195-216.

[5] E. Kalliamvakou, G. Gousios, K. Blincoe, L. Singer, D. M. German, and D. Damian, “The promises and perils of mining github” in Proceedings of the 11th working conference on mining software repositories, 2014, pp. 92-101.

[6] M. Malcher and D. Kuhn, Pro Oracle Database 18c Administration: Manage and Safeguard Your Organization's Data. Springer, 2019.

[7] G. C. Murphy, M. Kersten, and L. Findlater, “How are java software developers using the elipse ide?" IEEE software, vol. 23, no. 4, pp. 76-83, 2006.

[8] L. Filion, N. Daviot, J.-P. Le Bel, and M. Gagnon, “Using atlassian tools for efficient requirements management: An industrial case study” in 2017.

[9] P. Mcnamaara, “Ddos attack against bitbucket darkens amazon cloud," Network World, 2014.