Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 231 (2024) 421–426

Procedia Computer Science

www.elsevier.com/locate/procedia

The 4th International Workshop on Healthcare Open Data, Intelligence and Interoperability (HODII)
November 7-9, 2023, Almaty, Kazakhstan

# Low Code Approach for Business Analytics

Daniel Sá[a]*, Tiago Guimarães[a], Antonio Abelha[a], Manuel Filipe Santos[a]

[a]*ALGORITMI/LASI Research Center, University of Minho, Guimarães, Portugal*

**Abstract**

Low-code is an approach to programming that seeks to enable the creation of artefacts by both programmers and non-programmers. It enables the rapid creation and delivery of applications with minimal effort to write in a programming language as well as for installation and configuration of environments, training and deployment. This allows the development to follow the needs of a company in a faster and more efficient way.

Thus, a web platform was created that allows the user to customize a dashboard composed mainly of graphs and widgets, in order to make the most of the data. However, beyond this specific application, the platform can be applied to any other area with minimal effort from the developers.

The platform was developed using the technologies that are most suitable for the development of web applications. Thus we have the back-end that was developed using the Node.js platform, which uses one of the fastest growing languages in web development, JavaScript. For the interface was used the framework also in JavaScript, Vue.js. As for data storage, the non-relational data model was used and for its management and observation, the MongoDB software was used.

© 2024 The Authors. Published by Elsevier B.V.
This is an open access article under the CC BY-NC-ND license (https://creativecommons.org/licenses/by-nc-nd/4.0)
Peer-review under responsibility of the Conference Program Chairs

*Keywords:* Low-Code Approach; JavaScript; MongoDB; Vue.js.

* Corresponding author. Tel.: +351 253510319; fax: +351 253510300.
*E-mail address:* a85303@alunos.uminho.pt

1877-0509 © 2024 The Authors. Published by Elsevier B.V.
This is an open access article under the CC BY-NC-ND license (https://creativecommons.org/licenses/by-nc-nd/4.0)
Peer-review under responsibility of the Conference Program Chairs
10.1016/j.procs.2023.12.228

## 1. Introduction

Low-Code is an approach to programming that aims to enable the creation of artefacts by both programmers and non-programmers. This approach enables the creation and rapid deployment of applications with minimal effort in writing code in a specific programming language, installing and configuring environments, training and deployment. In this way, development can follow the needs of the business more quickly and efficiently [1].

These are high-growth approaches in the market, and there are currently many alternative technologies and closed solutions, as well as bespoke approaches designed to facilitate development and integration with more 'traditional' solutions [2].

It is imperative to design and develop solutions that reduce the cost and time of development, ensuring their adaptability and dynamism, either by adopting a completely Low-Code approach, using technologies and solutions already on the market, or by adopting a hybrid aspect, integrating approaches that guarantee a more Low-Code development on "traditional" programming platforms [1].

Application of a Low-Code approach to conduct proof of concept in diverse use cases within web development, particularly in healthcare.

## 2. Background

### *2.1. Low-Code*

Low-code development platforms provide technology mechanisms that facilitate and automate the development of software applications, in order to support the current needs of companies and to promote the digital transformation. This allows companies to develop applications without complex engineering and simplify their configuration, achieving speed and agility. In addition, these platforms offer a more cost-effective way to meet internal market and/or business requirements [2].

This type of development is on the rise and combines source code with interactive graphical interfaces, allowing you to quickly develop and deploy an application.

In general, Low-Code platforms are based on a graphical interface where the user only needs to design the aesthetics and functionality of the application, without having to resort to traditional programming techniques, thus avoiding the development of code syntax and solving implementation problems. Low-code development techniques are derived from the combination of fourth-generation programming languages (4GL) with rapid application development (RAD) concepts [1].

### *2.2. Node.JS*

Node.js is an open-source software platform based on the JavaScript programming language for developing server-side applications. According to Tilkov and Vinoski (2010), it is based on the runtime of a Google implementation, the "V8" engine [3].

It has become increasingly popular, especially when used in real-time applications where scalability and high performance are required. Unlike other technologies, it does not rely on multithreading, which means that it allows applications to run asynchronously in a single thread. This makes it suitable for applications with a large number of requests and connections running concurrently, because while one I/O (input/output) request is being processed, Node.js is still able to respond to and execute other requests [3].

### *2.3. Vue.js - JavaScript Interface Framework*

Today, most websites use JavaScript to present more interactive pages or to handle some specific functionality. Traditional web pages are characterized by being a multi-page application, where each time the user changes pages, a new HTML document is loaded. However, development models based on Single-Page Application (SPA) are increasingly being used, meaning that only the parts of the page that have changed are loaded by the server, thus reducing the waiting time when something new needs to be loaded onto the page and consequently improving the user experience [4].

Unlike other frameworks, Vue.js is a bottom-up, incremental framework developed by Evan You. It was developed because, as a Google employee, he felt he was repeating the same HTML code too often, so he started looking for tools to optimize his time. He couldn't find a framework or library that met his needs. He then started to develop a library that would allow him to develop applications quickly and also easily create and reuse components throughout the application [4].

Like React.js, it uses a virtual DOM, meaning that an ideal representation of the user interface is stored in memory before being synchronized with the real DOM. It focuses on a central library, but it is possible to add libraries as needed by the user [5], [6].

Vue.js automatically tracks JavaScript state changes and efficiently updates the DOM when changes are made, so it can be concluded that Vue, like React, is both reactive [7].

## 3. Low Code Systems

Low-code technology has been used in diverse areas such as manufacturing, customer relationship management (CRM), human resources, project management and accounting.

Some of the most popular Low-Code platforms are OutSystems, Microsoft PowerApps, Mendix, Salesforce, Appian, Zoho Creator and others. These platforms provide visual development environments that allow companies to quickly develop software with minimal programming and code development, reducing the need for users to have extensive programming experience. These platforms provide entry-level code, scripts and integrations so companies can prototype, build or scale applications without developing complex infrastructure [1]. Programmers and non-programmers can use these tools to practice rapid application development with customized workflows and functionality [1], [8], [9].

## 4. Architecture for Low Code Approach in Business Analytics

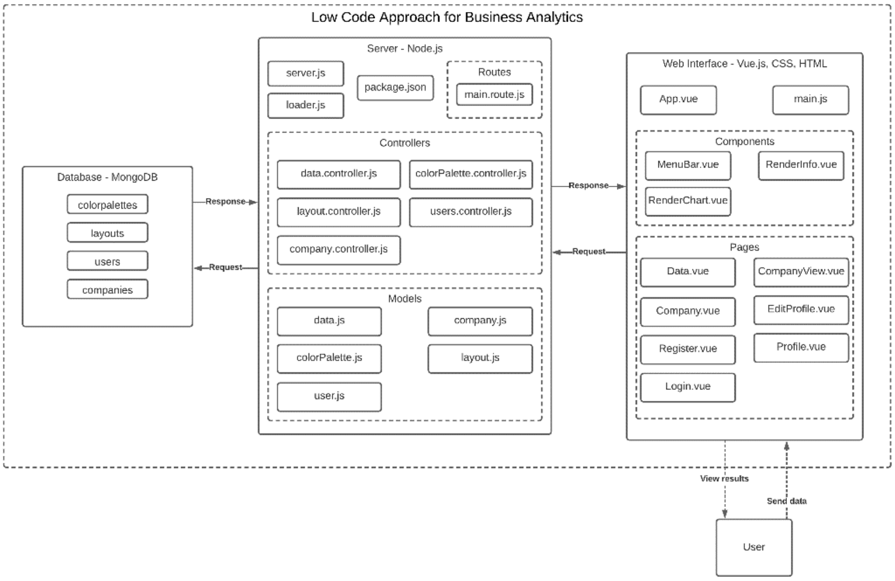


Fig. 1. Architecture for Low Code Approach in Business Analytics

In order to implement the proposed project, an architecture has been designed to answer the problem presented. The architecture, as shown in Fig. 1, can be categorized into three main components: the MongoDB-based database, the Node.js-based JavaScript server, and the web interface constructed using Vue.js, CSS, and HTML. This architecture also represents the process from user interaction with the platform to database requests.

The Mongo database consists of four basic tables: the "colourpalettes" table, responsible for storing the information related to the predefined color schemes and also customized by the users. Table "layouts" is responsible for storing all the layouts of each user, in other words, it is the table that contains the values of each component created in the dashboard, containing the graphics and widgets that a particular user adds to the dashboard of a company. The "Users" table is responsible for storing the information relating to each user. The "companies" table is responsible for storing information about each company. This database receives and responds to requests from the server.

The server was developed and configured in Node.js, is responsible for receiving and responding to all requests coming from the web interface, as well as making requests and receiving responses from the MongoDB database. This server uses the files shown in Fig. 1, where they are divided in groups, of which it contains the "Routes" group, characterized by only one file responsible for managing and forwarding the requests made to the server for each of the respective functions. The “Controllers” group contains all the functions that process and respond to events. The "Models" group contains how the data is stored in the database. The "server.js" file is the file responsible for running and starting the server, as well as accessing the database, which in turn accesses the "loader.js" file to load all the necessary dependencies and libraries before the server is ready to respond to requests from the web interface. Finally, the “Package.json” file corresponds to all the dependencies used by the server and their versions.

The web interface, developed in Vue.js, CSS and HTML, is responsible for the visual and interactive part of the project, being in charge of showing the user an interactive interface where he makes requests and receives responses from the server. This interface is made up of two files needed to start it, the "App.vue" and the "main.js". Then it has the "components", which are reusable instances of Vue.js. That is, they allow the programmer to reuse code without having to repeat it, as is the case with graphics, for example. Finally, we have the "pages", which are the pages displayed to the user, which in turn integrate the components.

## 5. Results

A web platform has been developed that allows the user to add, remove and edit a dashboard composed of graphics and widgets in a simple and intuitive way, thus allowing to create a dashboard that responds to all user needs without having to create or edit code. This platform can also be used in other areas without the need for major changes by the developer.

### 5.1. Dashboard

Fig. 2 presented bellow shows a set of customized graphics and widgets, selected and added by the user in a few clicks. The purpose of this page is to demonstrate the key points defined by the user for this specific company.

In a simple and intuitive way, the user can add new charts and widgets to this dashboard, as well as edit the color scheme used and share this layout (characterized by the set of charts and widgets) with any of his companies, all through the "Dashboard Options" button located in the upper right part of this dashboard, as shown in Fig. 3.

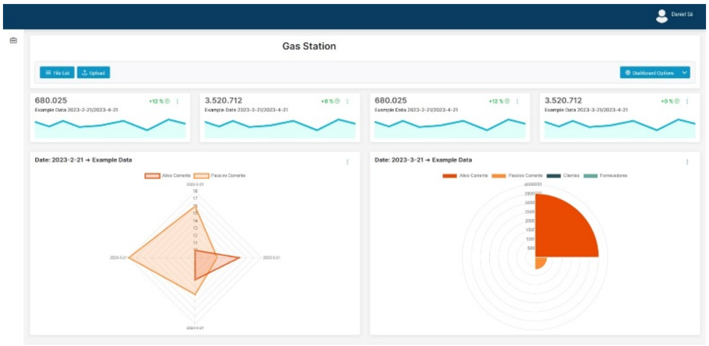

Daniel Sá
Gas Station
File List
Upload
Dashboard Options
680.025
+12 %
Example Data 2023-2-21/2023-4-21
3.520.712
+0 %
Example Data 2023-3-21/2023-4-21
680.025
+12 %
Example Data 2023-2-21/2023-4-21
3.520.712
+0 %
Example Data 2023-3-21/2023-4-21
Date: 2023-2-21 → Example Data
Ativo Corrente
Passivo Corrente
Date: 2023-3-21 → Example Data
Ativo Corrente
Passivo Corrente
Clientes
Fornecedores

Fig. 2. Dashboard Customised by the User

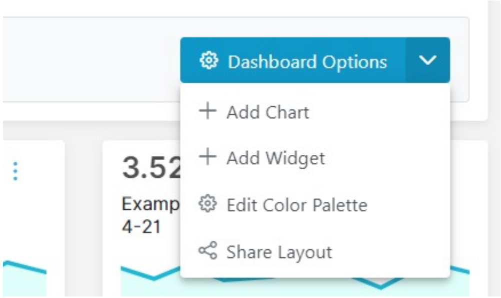


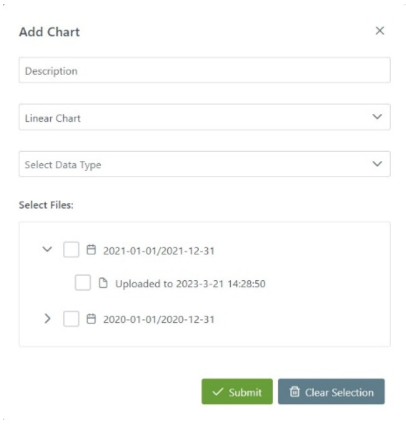
Fig. 3. Dashboard Options Menu

The "Add Chart" option from the menu shown in opens a new modal window with several options. In the first, it is possible to add a specific name for this chart, or to leave this option empty and the name of the chart will be filled in automatically according to the data selected for this chart and the date of the data.

The second option allows the user to select the type of chart to be added, with the following options: Linear chart, Bar chart, Pie chart, Doughnut chart, Polar area chart, Radar Chart. Once the chart type has been selected, a new field will appear allowing the user to select the data source to be used. This option only appears after the chart has been selected, because some of the charts available are able to display data based on several data sources, namely the "Linear Chart", the "Bar Chart" and the "Radar Chart", while the others can only accept one data source, as we can see in Fig.  and Fig. .

Finally, it can be select the type of data to be displayed. Based on the data in the database, a list is created with all the values available in each of the datasets.

As with charts, it is possible to add widgets to the dashboard and the method of adding them is identical to that of adding charts, except for the selection of the dataset to be used. For widgets it is necessary to select two datasets in order to make a comparison between them, taking into account the type of data selected. It can be seen in Fig. .


Add Chart

Description

Linear Chart

Select Data Type

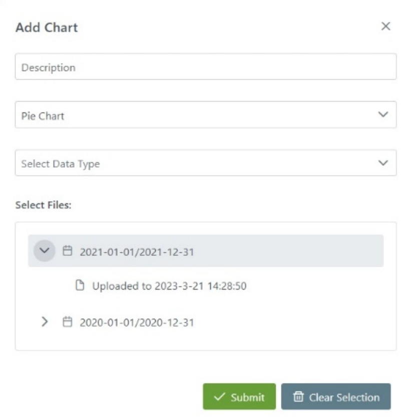
Select Files:

□ 2021-01-01/2021-12-31

□ Uploaded to 2023-3-21 14:28:50

□ 2020-01-01/2020-12-31

Fig. 4. Menu when the selection of the chart type is "Linear chart", "Bar chart" or "Radar chart".


Add Chart

Description

Pie Chart

Select Data Type

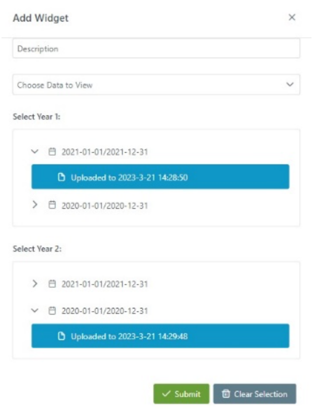
Select Files:

2021-01-01/2021-12-31

Uploaded to 2023-3-21 14:28:50

2020-01-01/2020-12-31

Fig. 5. Menu when the chart type selection is "Pie Chart", "Doughnut Chart" or "Polar Area Chart".


Add Widget

Description

Choose Data to View

Select Year 1:

2021-01-01/2021-12-31

Uploaded to 2023-3-21 14:28:50

2020-01-01/2020-12-31

Select Year 2:

2021-01-01/2021-12-31

2020-01-01/2020-12-31

Uploaded to 2023-3-21 14:29:48

Fig. 6. New widget creation menu

As we can see in Fig. 8, it is also possible to edit the color scheme, in this option a new menu appears where it allows the user to edit the color scheme for the graphics, it can select and customize each of these predefined schemes, as it can be seen in Fig. . In addition, all the charts have an option that allows the user to customize this color scheme, as well as the data that the chart displays, as it can be seen in Fig. . In addition to the graphs, it is also possible to edit the widgets at any time.

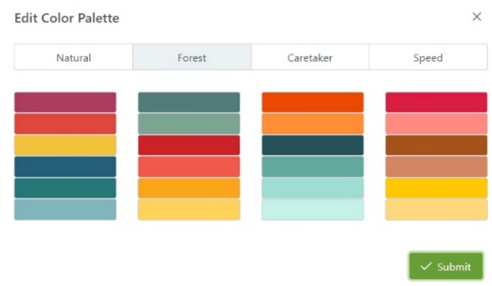
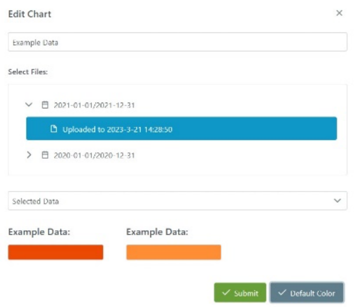

Edit Color Palette

Fig. 7. Menu with predefined colour palettes


Edit Chart

Example Data

Select Files:

2021-01-01/2021-12-31

Uploaded to 2023-3-21 14:28:50

2020-01-01/2020-12-31

Selected Data

Example Data: 🟥

Example Data: 🟧

Fig. 8. Example of editing a doughnut chart, where it is possible to see the individual colour edits.

Finally, as it can be seen in Fig. 8, it is possible to share this layout with other dashboards, that is, if the user wishes, he can have a set of dashboards representing different entities, such as people, companies, objects, among others. This sharing allows the user to apply the same analyses to different content.

## 6. Conclusion

In this study, the focus has been on the development of a Low-Code approach to business analytics, which has resulted in the platform described and demonstrated in the architecture and in the results. This application allows users to configure and customize a panel, made up of graphs and widgets, that allows the analysis of data uploaded by the user in a personalized way, without having to resort to any type of code.

For future work, it is expected that this platform will be improved in order to use AI algorithms to make even better use of the data, as well as allowing the user greater customization of the types of graphs and the ability to add small notes to the dashboard that may become relevant in the analyses.

**Acknowledgements**

This research was funded by Fundação para a Ciência e Tecnologia, within the Project Scope: UIDB/00319/2020.

## References

[1] R. Waszkowski, “Low-code platform for automating business processes in manufacturing,” *IFAC-PapersOnLine*, vol. 52, no. 10, pp. 376–381, Jan. 2019, doi: 10.1016/J.IFACOL.2019.10.060.

[2] R. Sanchis, Ó. García-Perales, F. Fraile, and R. Poler, “Low-Code as Enabler of Digital Transformation in Manufacturing Industry,” *Applied Sciences 2020, Vol. 10, Page 12*, vol. 10, no. 1, p. 12, Dec. 2019, doi: 10.3390/APP10010012.

[3] S. Tilkov and S. Vinoski, “Node.js: Using JavaScript to build high-performance network programs,” *IEEE Internet Comput*, vol. 14, no. 6, pp. 80–83, Nov. 2010, doi: 10.1109/MIC.2010.145.

[4] “Learning Vue.js 2 - Olga Filipova - Google Livros.” Accessed: Jun. 15, 2023. [Online]. Available: https://books.google.pt/books?hl=pt-PT&lr=&id=nszcDgAAQBAJ&oi=fnd&pg=PP1&dq=vue.js&ots=9oMhJeTdlQ&sig=nRCyZGe97LwTb0abNfi4cEhtd8E&redir_esc=y#v=onepage&q=vue.js&f=false

[5] “Virtual DOM and Internals – React.” Accessed: Jun. 16, 2023. [Online]. Available: https://legacy.reactjs.org/docs/faq-internals.html

[6] “Vue.js: Construa aplicações incríveis - Caio Incau - Google Livros.” Accessed: Jun. 16, 2023. [Online]. Available: https://books.google.pt/books?hl=pt-PT&lr=&id=Ft-8DgAAQBAJ&oi=fnd&pg=PT2&dq=vue.js&ots=_acX7_L03x&sig=VYLZJ-cj9Cuwyu-R4AvI7b783kc&redir_esc=y#v=onepage&q=vue.js&f=false

[7] “Introduction | Vue.js.” Accessed: Jun. 16, 2023. [Online]. Available: https://vuejs.org/guide/introduction.html

[8] A. Sahay, A. Indamutsa, D. Di Ruscio, and A. Pierantonio, “Supporting the understanding and comparison of low-code development platforms,” *Proceedings - 46th Euromicro Conference on Software Engineering and Advanced Applications, SEAA 2020*, pp. 171–178, Aug. 2020, doi: 10.1109/SEAA51224.2020.00036.

[9] P. Vincent *et al.*, “Magic Quadrant for Enterprise Low-Code Application Platforms Critical Capabilities for Enterprise Low-Code Application Platforms View All Magic Quadrants and Critical Capabilities Strategic Planning Assumption,” 2020.