Journal of Current Research in Engineering and Science
Bi-Annual Online Journal (ISSN : 2581 - 611X)
Volume 5- Issue 1, Paper 7 January 2022

Design and Development of Analytical Dashboard

N Amar, Shwetha Baliga
Electronics and Communication Dept
R V College of Engineering Bengaluru, India.
namar.ec18@rvce.edu.in, shwethaprabhun@rvce.edu.in

***Abstract*—in the present scenario organizing the data available to an organization is one of the most important thing to do. This data circulated from the outer world is used in managing some of the most valuable assets of a company therefore techniques of data organization is very important. Getting insights out of this data could help one obtain better business intelligence and play a major role in your company's success. As every organization is dependent on managing the data being created for its processes, and with an increase in the complexity of operations, there is always a strive to simplify processes or to automate the process of managing the data.**

So as a part of this project and to hold the accounts of all the registered devices of the company, a dashboard is built which will be the interface that the organization needs. All the required backend services are also developed and implemented. This dashboard shows the statistics and overview of all the company's devices that are already registered. It also holds a place for all the geographical areas where the devices are located. Total no of customers and their respective devices is also registered in this dashboard. Moreover one can also add the information of the products that are newly released to the database very easily. One of the best features of this dashboard is that organization can send firmware updates related to a particular product to the user in a very efficient way. This dashboard will also record all the reviews of a particular product which will then be classified into good, bad in the terms of UI, services and IOT so that people of the organization can work on the backlogs more efficiently. In addition that, a thought of inserting images of each devices are also implemented which will prove to be helpful in identifying the product more efficiently and understanding the problem related to the device will also be easier. Having all of the above mentioned functionalities about various devices will be a great support to the organization and will be helpful in managing the devices. The UI part and various screens of the dashboard is shown in the results and discussion part for the reference.

I. Introduction

A dashboard is a visual display of data. A dashboard is necessary of an organization for analyzing the activities and growth of the company. Although it has a wide range of ap- plications, its main purpose is to present information quickly, like Key Performance Indicators (KPIs). The information for a dashboard often comes from a linked database and is shown on its own page. As they send managers and clients regular emails and notifications about the status of their projects, dashboards provide improved customer service through monitoring.

Dashboards provide a comprehensive picture of the com- plete organisation by giving the manager a bird's eye view of the performance of sales, data inventory, site traffic, social me- dia analytics, and other related data that is visually presented on a single dashboard. It is also possible to roll up data in order to give a unified perspective throughout an organization.A dashboard for the presentation of marketing data makes the process of marketing easier and more dependable as compared to doing it manually, dashboards contribute to improved man- agement of marketing/financial plans. Many firms rely heavily on web analytics to shape their marketing strategies. Because the data is more accurate and in one place, dashboards also make it easier to track sales and provide financial reports.

II. Literature Review

The basis of REST is discussed in this paper [1]. It provided a brief comparison of RESTful services and traditional SOAP web services and discussed ways to make it easier for users to find the services we create. Construction of a standard online API that guides you through constructing the application using human-readable URL and URI parameters A full-fledged RESTful service implementation was also covered, with a focus on creating a production-ready RESTful service that leverages NoSQL to store its data. choosing an acceptable authentication strategy to secure the application, which in- cludes limiting access to your data. This paper [12] presents how to build a complete data-driven website. Almost every component of a typical large-scale web development project will be included in the website we build. The project will be created with the well-known Node.js Express framework, and it will use MongoDB to persist data. The foundational steps necessary to set up the server's core and begin serving content were covered in the first few chapters. This includes setting up the environment to start using Node and MongoDB as well as a short introduction to the fundamental ideas behind both technologies. Then it covered creating a web server from scratch that uses Express JS and will take care of supplying all required files. This paper [4] explains the idea of

Journal of Current Research in Engineering and Science
Bi-Annual Online Journal (ISSN : 2581 - 611X)
Volume 5- Issue 1, Paper 7 January 2022

model-driven development and suggests the UML profile for AngularJS and supporting transformation templates that may be used to model and create code for AngularJS applications, respectively. For further work, a more thorough assessment is planned for applications with varying sizes and levels of complexity. Support for modelling applications in a particular area would be an addition to the suggested UML profile. This paper [6] discusses the fundamentals of how the webserver and browser communicate as well as the functionality needed to run today's websites. Node.js package modules offer features that node.js does not have by default. These modules are available for download from the NPM repository, and you may also write your own and submit it. It also discusses the various ways you may add tasks to the Node.js event queue. Either directly

schedule work or use event listeners or timers to add work. Additionally, it demonstrates how to add events to your own unique modules and objects. The main focus of this paper [13] is on Visual Studio's conversion from ADAL to MSAL, which has improved our ability to implement Conditional Access, Multi-factor Authentication, and other new AAD capabilities that benefit our clients. The.NET Core SDK and Visual Studio 2019 can be used to execute this task. The idea is to keep an app's functionality and authentication technique distinct. Azure Active Directory (Azure AD) controls the login process to prevent sensitive information (like passwords) from falling into the wrong hands at websites and applications. Because of this, identity providers (IdPs) like Azure AD can offer smooth single sign-on experiences, let users login using methods other than passwords (a user's phone, face, or other biometrics), and reject or prioritise authentication attempts.

III. FUNDAMENTALS

A data dashboard is a visual representation of the most cru- cial information required to accomplish one or more goals. The data is gathered and arranged on a single screen so that it can be watched quickly.Graphs and other visual representations of data are frequently used in data dashboards, along with a minimum amount of text to describe the signs that are shown there. The user may directly compare and draw conclusions from the data by having these visualisations displayed on a single screen, which is not possible if the data is split across multiple displays or needs scrolling to view.

When properly created, dashboards allow data repositories to be converted into usable information. The dashboard's information is typically used to inform decisions and actions, and the visualisations make it easier to spot trends and patterns.Many dashboards are web-based, so anyone with an internet connection can access them. While certain data dash- boards can be accessed by anybody, others are only available to those who have been granted access. The dashboard designd in this project is an organizational dashboard so only the employees have the access to it.

A. Authentication

In order to gain access to an application, API, microservices, or other piece of data, authorised users must first authenticate their identities. Authorization is the process of giving a verified user access to carry out a certain activity on a set of resources. To deal with sensitive data assets, authentication and au- thorisation are both necessary. Without any of them, the data will open to unwanted access and data breaches. To keep their data secure, every organisation strives to employ the best authentication procedure[7]. To verify user identification, a variety of authentication procedures can be utilised. Several of them are listed below.

- **Single Sign-On (SSO):**Through SSO, users can log into many applications using just one set of login information. When a user logs into a distributed across multiple domains using SSO, a federation is used. **Multi-Factor Authentication (MFA):**Different methods of authentication are used with Multi-Factor Authenti- cation (MFA). The user is required to enter a one-time access code that the website delivers to their mobile phone in order to log in using their user name and password. In order to increase security, it offers a higher level of assurance throughout the authentication phase.
- **Consumer Identity and Access Management (CIAM):**This system offers a number of capabilities, including client registration, self-service account management, consent and preference management, and other authentication features.

In this project single sign-on is implemented using the Microsoft authentication library.

B. Microsoft Authentication Library(MSAL)

Developers can obtain tokens from the Microsoft identity platform to authenticate users and access secured web APIs by using the Microsoft Authentication Library (MSAL). Secure access to Microsoft Graph, other Microsoft APIs, third-party web APIs, or your own web API can all be provided using it. MSAL is compatible with a wide range of application platforms and architectures, including.NET, JavaScript, Java, Python, Android, and iOS.

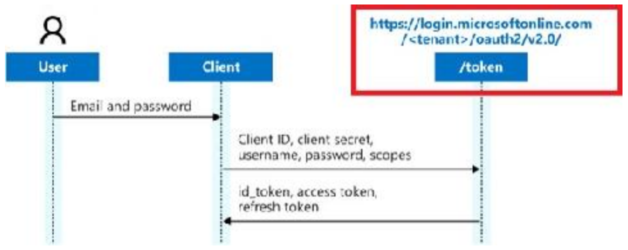


Fig. 1. Workflow of Microsoft authentication library [13]

The different types of tokens received in 1 are:

- **Access Token:**An access token is a piece of information that will include in the Authorization header of your request and that the API you are accessing can verify and use to allow you access. Access tokens typically come in JWT format, but they are not required to. Access tokens are typically only valid for a brief period of time because losing one essentially means losing the keys to whatever it is that the access token is good for (30mins-1hr etc.). Although challenging, centrally revoking access tokens is not impossible.
- **ID Token:**The user's identification is represented by an ID Token, which is typically in JWT format but need not be. An ID token is only used to identify the user; it cannot contain any audience or authorization information[17].
- **Refresh Token:**Refresh tokens are durable tokens that are used to obtain fresh access tokens. It typically receive an audience, also known as an access token, for a specific resource. Refresh tokens should only be used by clients who can securely store them.

IV. REST ARCHITECTURE

To create the backend API services an architecture which is compatible with the requirement must be followed. Since the REST architecture had many advantages over other architec- ture, this was implemented.The term REST stands for REpre- sentational State Transfer, and it refers to a type of software architecture that is becoming popular in web service[3]s. It has been created with simplicity, scalability, and generality in mind and is aimed at distributed information systems. The resource- oriented architecture of REST APIs sets them apart from conventional local API design in a number of ways.Distributed information systems are the focus of the REST software architectural style.In other words, it is designed for systems where every component must communicate across a network even when they are not all located on the same physical machine.

In typical local APIs, calling hard-coded functions related to the component being talked with is often how API queries are made. A component can start a conversation by calling a particular function in another component, and that component can then answer by returning a value. Instead, requests in REST are built as self-descriptive messages and submitted through a standard uniform interface.

A. Client-server

It is best to divide the functions of a client and a server. As a result, the client is more portable because it no longer needs to worry about server-specific issues like data storage. As a result, the server becomes simpler and more scalable because it no longer needs to worry about the user interface. Compo- nents can develop independently thanks to this separation of concerns, which is another advantage[15].

B. HTTP

The World Wide Web uses the Hypertext Transfer Protocol (HTTP), an application protocol for distributed hypermedia systems. Since Roy Fielding introduced REST and contributed to the development of the HTTP specification, the two pro- tocols have a shared history. When utilised properly, HTTP can be used to construct RESTful APIs because it implements many of the REST requirements. Even though HTTP is a web- based protocol, many of its principles ought to be transferable to local settings.

C. Uniform Interface

To make the overall system design simpler, components should have a consistent interface. For this, REST specifies four interface requirements. Because information must be conveyed in a standardised format rather than an improved application-specific format, uniformity may actually reduce efficiency[8].

- **Identification of resources:** Each distinct resource has an identification to help distinguish it during component interactions. An example of this is a URI[9]. The HTTP protocol uses this principle to identify all resources,including web pages, photos, and other documents, using unique URLs.
- **Manipulation of resources through representations:** By providing representations, clients can edit and add resources on the server. A client produces and transmits a request when it wishes to, for example, create, update, or delete a resource on a server creating representation of the resource in the desired state.
- **Self-descriptive messages:**In addition to data, messages between components also include metadata that specifies how to interpret their contents. The URI of the resource that the representation is representing or the nature of the representation, such as the file format it is in, are examples of typical metadata[2]. Control data defining the purpose of a communication is also included in messages. A client may define the action it wishes to do on a resource, such as whether it wants to add a new resource, update an existing resource, delete a resource, etc.
- **Hypermedia As The Engine Of Application State (HA- TEOAS):**Except for a known initial resource identifier, the client only changes states by choosing between state transitions provided by the server in received represen- tations. This is how the user interface of a web browser usually works; a user navigates to a known bookmark and then navigates between web pages and performs actions by clicking on links or submitting forms provided by the web server.

V. Front End Development

Front end development is the branch of web design that concentrates on user interface design. It entails converting the back end developers' code into a graphical interface while ensuring that the data is displayed in an understandable man- ner.For creating intricate user interfaces, front-end frameworks are an effective tool[16]. It supports the development of a freestanding, maintainable, modular architecture that makes it simple to create apps and work with other developers[10]. Frontend for the project is developed using tools such as HTML, CSS, Bootstrap, JavaScript and Angular.Js. There are total four features which are created for the project which is discussed further.

- **Login:**It is the first page which is displayed to the employees where they enter their employee ID and email ID. Since there is MSAL login service, the user is authenticated by Microsoft and will go to next page only for the organization employees.
- **Homepage:** The homepage of the dashboard consists of information regarding the products and the customers. It has the statistics of number of customers registered and can be sorted according to the year, month and the region. It also shows the number of devices connected according to the geographical location. There is a pie chart available to show the graphical representation of registered devices in each state.
- **OTA:** OTA stands for On The Air feature which is used to deploy any firmware update to the product directly. It

also has all the information of the firmware versions of all the products. It is very useful to update the firmware from the cloud to the end user.

- **Add Product:** This feature is used to add a new products information t the database.It has several fields of informa- tion such as brandName, title, subtitle, description, model name, model number, product code, product type, etc.
- **Upload Image:** The images of all the products will be uploaded using this feature. The uploaded images will be stored in bobstorage along with the path. Only the images of existing productcode can be added.

VI. Results Discussions

The results of the dashboard design are discussed in this section. The outcomes are shown in two phases. The user interface is the first phase, and the second process is the backend services required for the UI to function properly.

A. Web UI

The user interface is developed using angular.js framework and total four components are added. Detailed discussion of each component result is in this section.*Backend Services*

Several APIs are developed to support the functionality of the dashboard website. Using these APIs a connection of client to the server is made and the client's data is stored in mongoDb database. These results are further discussed in this section.

Journal of Current Research in Engineering and Science
Bi-Annual Online Journal (ISSN : 2581 - 611X)

Volume 5- Issue 1, Paper 7 January 2022

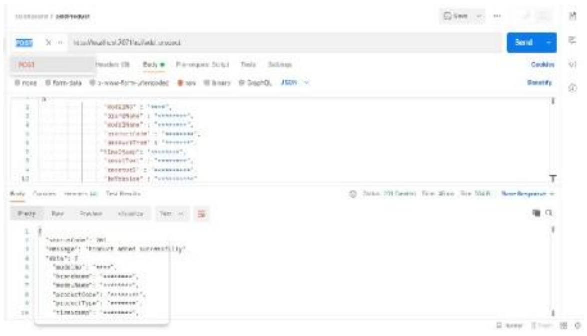
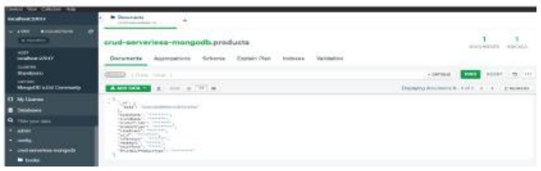


Fig. 4. API responses in Postman application

1) *API Responses:* An HTTP request method is developed for each functionality of the UI and is integrated with the frontend code. Whenever a feature in the UI is accesed, a background API will run by taking the request and will provide a HTTP response in return. This response can be seen in the figure 4.



1) *Login:* The login page of the dashboard is shown in Fig 2. This is the first component shown as soon as anyone opens the dashboard website. The user has to login with his credentials and the MSAL service will authenticate the user and will be redirected to next page only if the user is the employee of the organization[14].

2) **Fig. 2. Login page**

3) *Add Product Component:* The user interface of updating the firmware is designed as shown in the figure3. The list of products and their respective firmware will be visible as soon as this component is opened. The product code must be chosen whose firmware needs to be updated.

2) *MongoDb Database:* All the data of dashboard is stored in the mongoDb database as shown in the figure5. This is the data of a product stored. It stores the data in javascript object notation(JSON) format.

VII. Conclusion and Future Scope

A. Conclusion

As we already know that handling and organizing the data is one of the most important aspects as it enlightens us with new prespectives and ideas for our buiseness. Current project on this dashboard suggests that this will clear the major amount of clutter and save us time. Handling and updating the software of the device will be much easier as it is being done through this interface. Categorizing and classifying all the devices data will be very much useful for the company. This dashboard is a start to make all these processes bit more autonomous.

We have operated on various features of the dashboard which are suggested above and still there are ideas which are needed to be thought on and implemented. Firmware updates

or software updates related to the onboarded device or the de- vices registered can be updated through this dashboard easily and we can track all devices at once. We have also planned to include the fault analysis of the device in the dashboard so that we can send the notification to our customers beforehand for any faults or servicing required for the device they have onboarded. Earlier the software team of the company had to manage all devices separately so that was bit more tedious job to do. This dashboard will decrease those unnecessary efforts as we have made a single interface system to handle all the devices. As we are into the business of making more and products therefore this dashboard will be a huge help to the software team to manage all the devices in a much efficient way.

B. Future Scope

The future Scope of this dashboard lies in the pipeline that it is making to connect different devices in service[11]. Predictive Maintainaince is the another aspect of the future scope of the dashboard. So we are planning on introducing this in the dashboard which will help us to get track of the anomalies in the operation of the devices and possible defects in the equipment. We have also planned to map this dashboard with the company’s application and work on the behavioural maintainance[5]. This will require us to capture all the real time data coming from the dashboard so that we can collaborate with other teams to get some insights from that data to increase the quality of our devices. We will also work on sending notifications to the customers regarding the usage of the devices.

REFERENCES

[1] Valentin Bojinov. *RESTful Web API Design with Node. js 10: Learn to create robust RESTful web services with Node. js, MongoDB, and Express. js*. Packt Publishing Ltd, 2018.

[2] Stefan Bosnic, I sˇtvan Papp, and Sebastian Novak. “The development of hybrid mobile applications with Apache Cordova”. In: *2016 24th Telecommunications Forum (TELFOR)*. IEEE. 2016, pp. 1–4.

[3] Angelo P Castellani et al. “Web Services for the Internet of Things through CoAP and EXI”. In: *2011 IEEE In- ternational Conference on Communications Workshops (ICC)*. IEEE. 2011, pp. 1–6.

[4] Wutthichai Chansuwath and Twittie Senivongse. “A model-driven development of web applications using AngularJS framework”. In: *2016 IEEE/ACIS 15th In- ternational Conference on Computer and Information Science (ICIS)*. IEEE. 2016, pp. 1–6.

[5] Anjali Chauhan. “A Review on Various Aspects of MongoDb Databases”. In: *International Journal of En- gineering Research & Technology (IJERT)* 8.5 (2019).

[6] Brad Dayley. *Node. js, MongoDB, and AngularJS web development*. Addison-Wesley Professional, 2014.

[7] Xinyang Feng, Jianjing Shen, and Ying Fan. “REST: An alternative to RPC for Web services architecture”. In: *2009 First International Conference on future infor- mation networks*. IEEE. 2009, pp. 7–10.

[8] Brad Green and Shyam Seshadri. *AngularJS*. ” O’Reilly Media, Inc.”, 2013.

[9] Florian Haupt et al. “A framework for the structural analysis of REST APIs”. In: *2017 IEEE International Conference on Software Architecture (ICSA)*. IEEE. 2017, pp. 55–58.

[10] Andrew John Poulter, Steven J Johnston, and Simon J Cox. “Using the MEAN stack to implement a RESTful service for an Internet of Things application”. In: *2015 IEEE 2nd World Forum on Internet of Things (WF-IoT)*. IEEE. 2015, pp. 280–285.

[11] Dario Sabella et al. “Developing software for multi- access edge computing”. In: *ETSI white paper* 20 (2019), pp. 1–38.

[12] Mithun Satheesh, Bruno Joseph D’mello, and Jason Krol. *Web development with MongoDB and NodeJs*. Packt Publishing Ltd, 2015.

Volume 5- Issue 1, Paper 7 January 2022

[13]D Subbarao et al. “Microsoft Azure active directory for next level authentication to provide a seamless single sign-on experience”. In: *Applied Nanoscience* (2021), pp. 1–10.

[14] Mohamed Sultan. “Angular and the Trending Frame- works of Mobile and Web-based Platform Technologies: A Comparative Analysis”. In: *Proc. Future Technolo- gies Conference*. 2017, pp. 928–936.

[15] Erik Sundvall et al. “Applying representational state transfer (REST) architecture to archetype-based elec- tronic health record systems”. In: *BMC medical infor- matics and decision making* 13.1 (2013), pp. 1–25.

[16] Yongkang Xing, JiaPeng Huang, and YongYao Lai. “Research and Analysis of the Front-end Frameworks and Libraries in E-Business Development”. In: Feb. 2019, pp. 68–72. DOI: 10.1145/3313991.3314021.

[17] Xinshuang Zhang et al. “The implementation and ap- plication of the internet of things platform based on the REST architecture”. In: *2011 International Conference on Business Management and Electronic Information*. Vol. 2. IEEE. 2011, pp. 43–45.