Serverless Web Application: To Overcome Server Management Problem

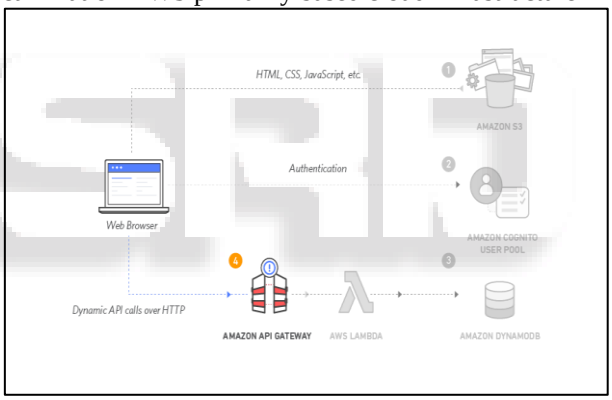
Jatin Shastri[1] Kapil Gupta[2] Mayank Gupta[3] Kavita Namdev[4] Ritesh Khedekar[5]
[1,2,3]Student [4]Sr. Assistant Professor [5]Assistant Professor
[1,2,3,4,5]Department of Computer Science & Engineering
[1,2,3,4,5]Acropolis Institute of Technology& Research College, Mangliya Square Indore-453771 Madhya Pradesh, India

Abstract— Serverless computing permits you to make and run applications and services without concerning about servers. Lately, the recognition and adoption of serverless computing or Function-as-a-Service are full-grown substantially, and it emerges as a higher thanks to manage value, responsibility, handiness, and quantifiability. This paper presents a detail of serverless offerings from leading cloud suppliers like AWS, Azure, Google Cloud Platform and some open supply. It compares them aspect by aspect within the relevant class like reason, storage, database, messaging, API management, and tooling. additionally provides comparative analysis on obtainable serverless architectures for the foremost common use cases among cloud provider's setting. it'll additionally emphasize on edges, open problems, potential solutions, and therefore the way forward for the technology. With serverless computing, your application still runs on servers, however all the server management is completed by AWS. victimization AWS and its Serverless Platform, you'll build and deploy applications on efficient services that give integral application handiness and versatile scaling capabilities. This permits you to concentrate on your application code rather than worrying concerning provisioning, configuring, and managing servers. Building a serverless application permits you to concentrate on your application code rather than managing and in operation infrastructure.

Keywords: Cloud Computing, Amazon Web Service, AWS Lambda, Dynamo DB, Command line interface, FaaS, VMs, intuitive, pay-as-you-go, Microservices, containerization, Serverless

I. INTRODUCTION

Our project is devoted towards serverless app design. As several organiztions tend towards a service homeward design, developers can usually ponder whether a lot of and a lot of components of their service might be moved into the cloud. Databases, file storage, and even servers are slowly transitioning to the cloud, with servers being run in virtual containers as opposition being hosted on a zealous machine. Recently, FaaS (function as a service) were introduced to permit developers to transfer their "application logic" to the cloud while not requiring the server, primarily abstracting the servers away. The serverless model conjointly offers many alternative blessings over ancient servers like prices and easy scaling. therefore would it not be doable to utterly replace the server with a serverless model? Taking under consideration the constraints of the FaaS model, we tend to began to make a totally useful, cloud primarily based, serverless app. Here the advantages of our application came. it's an internet application that is employed by user in addition as developers. Developers uses this application to deploy their net application that takes less time. At identical time users also can use net application to access the content. therefore there's no server management required by user in addition as developer. the appliance also will offer info concerning the sold out books therefore there {are also|also ar|are} records of best commerce books are accessible. a web store code comes that acts as a central info containing varied books available beside their title, author and value. This project may be a web site that acts as a central book store. so as to supply optimum performance and avoid web site crash the system must be hosted on a cloud infrastructure. The sql info stores varied book connected details. A user visiting the web site will see a good vary of books organized in various classes. The user could choose desired book and consider its value. The user could even explore for specific books on the web site. Once the user selects a book, he then must fill in an exceedingly type and therefore the book is set-aside for the user. The system therefore handles an outsized quantity of users with ease victimization AWS primarily based cloud infrastructure



II. LITERATURE SURVEY

A. BACKGROUND

1) Classic Web Application Architecture

A static resource is a file. for instance, it may be Associate in Nursing hypertext markup language, JPEG, PDF, or MP4 file that lives on the net server. The server can come back the document mere by the request in its response body.A dynamic resource is a resource that gets built by the server on the fly. Associate in Nursing example of a dynamic resource could be a search engine's search results page. Usually, the response body of a dynamic request are going to be formatted in hypertext markup language. once it involves internet applications, we have a tendency to trot out dynamic resources.

The net server is serving an internet application, and typically the net application contains a controller with the logic that routes the user's request to a particular action to perform on the server. Once the net server is finished process the user's request, the server sends a response back to the consumer within the style of an internet page

response. A server-side programing language (such as Go, Perl, PHP, Python, Ruby, and Java) is employed to method the requests sent from the net browser.

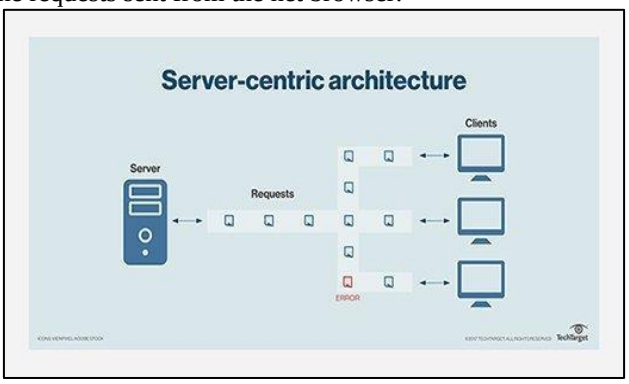


Fig. 2.1: Process flow of classic web application architecture

A key issue to creating an internet site friendly to go looking engines is discoverability. Besides having nice content, a hunt engine friendly web site additionally desires permalinks – internet links that ar meant to stay in commission for good. Descriptive and well-named URLs is registered as routes with the server-side's router. These routes find yourself serving as permalinks, that the computer programme larva crawlers will simply index whereas crawl through the web site.

2) Single-Page Applications (SPAs)

Instead of loading fully new pages from the server whenever for a user action, single page internet applications permits for a dynamic interaction by means that of providing updated content to this page. AJAX, a concise type of Asynchronous JavaScript and XML, is that the foundation for enabling page communications and thus, creating SPAs a reality. as a result of single-page applications forestall interruptions in user expertise, they, in a way, jibe ancient desktop applications. SPAs square measure designed during a manner so they request for many necessary content and knowledge parts. This results in the procural of associate degree intuitive furthermore as interactive user expertise. A single-page application (SPA) could be a web site style approach wherever every new page's content is served not from loading new markup language pages however generated dynamically through JavaScript's ability to control the DOM parts on the prevailing page itself. during a additional ancient web content design, an index.html page would possibly link to alternative markup language pages on the server that the browser can transfer andshow from scratch.

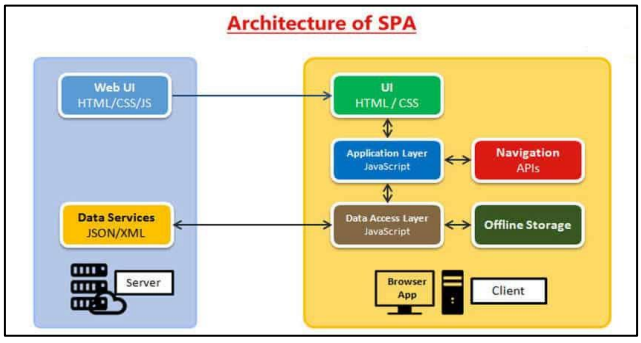


Fig. 2.2: Process flow of Single Page Application architecture

3) Microservices

These ar little and light-weight services that execute one practicality. The Microservices design framework incorporates a range of benefits that permits developers to not solely enhance productivity however additionally speed up the complete preparation method. The parts creating up Associate in Nursing application build mistreatment the Microservices design aren't directly enthusiastic about one another. As such, they don't necessitate to be engineered mistreatment constant programing language.Hence, developers operating with the Microservices design ar absolve to obtain a technology stack of alternative. It makes developing the appliance easier and faster. Microservices became massively widespread in recent years. Mainly, as a result of {they come|they ar available} with some of advantages that are super helpful within the era of containerization and cloud computing. you'll be able to develop and deploy every microservice on a distinct platform, mistreatment totally different programming languages and developer tools. Microservices use arthropod genus and communication protocols to act with one another, however they don't have confidence one another otherwise.

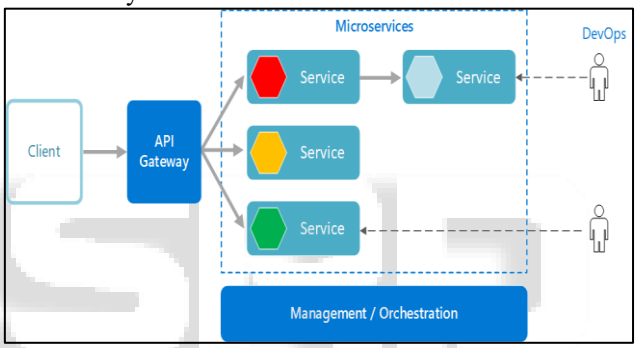


Fig. 2.2: Process flow of Microservices architecture

4) Proposed solution

Serverless computing offers variety of benefits over ancient cloud-based or server-centric infrastructure. for several developers, serverless architectures provide bigger quantifiability, additional flexibility, and faster time to unleash, all at a reduced value. With serverless architectures, developers don't got to worry regarding getting, provisioning, and managing backend servers.The project aims at the subsequent matters:

- No server management is necessary.
- Developers are only charged for the server space they use, reducing cost.
- Serverless architectures are inherently scalable.
- Quick deployments and updates are possible.
- Code can run closer to the end user, decreasing latency
- Advantages over traditional servers such as costs and ease of scaling.
- The code automatically scales up as needed.
- This system saves both time and travelling cost of customers.
- Reduce the investment necessary in DevOps, which lowers expenses

III. METHODOLOGY

Serverless design aims to unravel these issues. instead of having one machine to host employment, we will utilize

publically obtainable services, like Amazon net Services (AWS) Lambda functions and cloud hosted databases, to switch the standard server. This provides North American nation with many key advantages, Since AWS offers "Managed Services", you not would like a full team of system directors to line up the low-level aspects of your system. Amazon takes care of that for you as a core side of their platform. By employing a serverless design you not ought to invest an oversized amount of cash up front on a physical machine. Rather AWS offers a pay-as-you-go model for all of their hosted services. because the demand on your hosted service will increase, AWS mechanically provisions the resources needed to satisfy the demand. you'll be able to proportion throughout high volume periods and scale down throughout low, all the whereas solely paying for the resources you truly use. This, in effect, offers you unlimited capability

System Design Diagrams

1) Use case

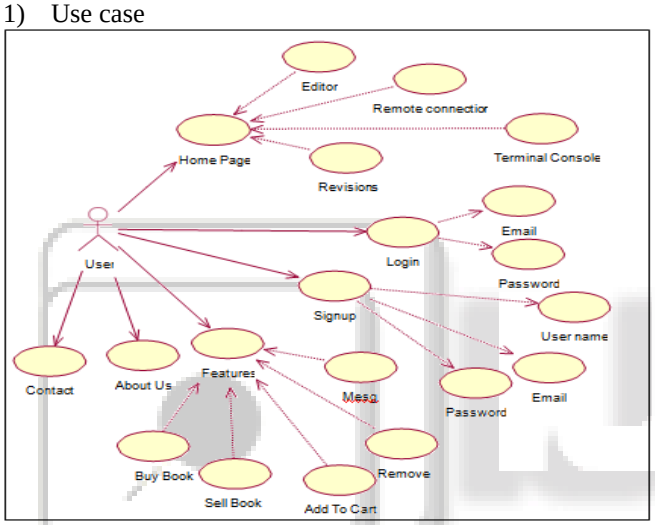


Fig. 3.1: Use Case Diagram

2) Sequence diagram

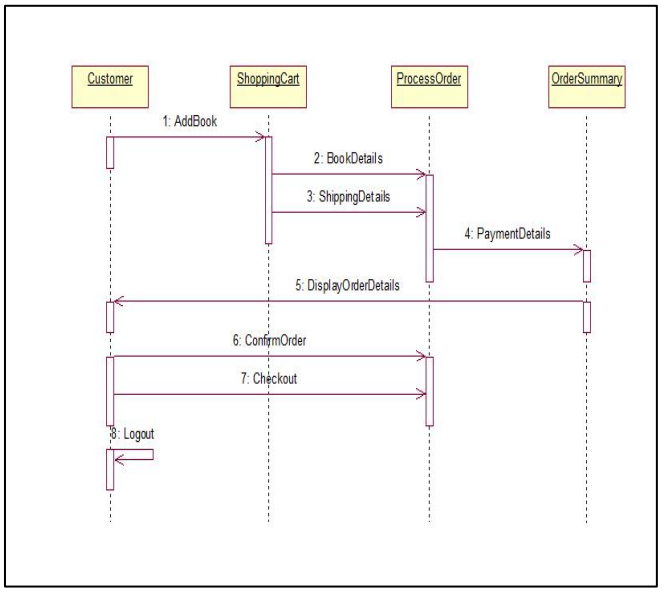


Fig. 3.2: Sequence Diagram

3) Class Diagram

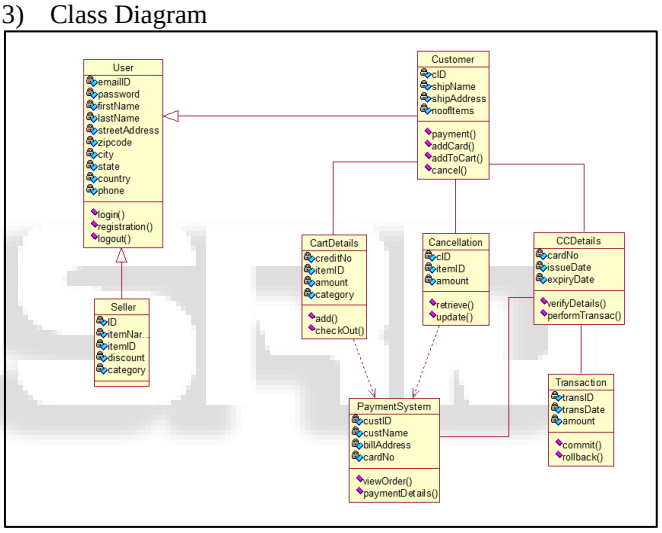


Fig. 3.3: Class Diagram.

4) Data Flow diagram

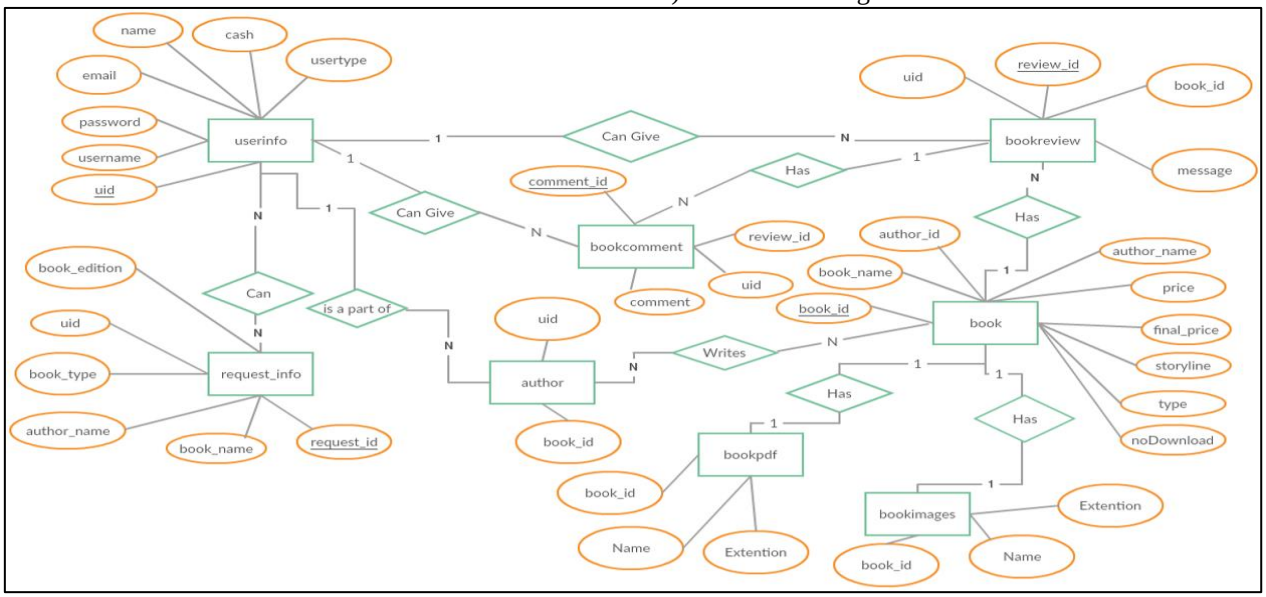


Fig. 3.4: DFD

IV. RESULTS

Our project would be really helpful for organizations to run their application without the management of servers. And it will be able to host web application efficiently without any delay and will be highly scalable and fault tolerant.

V. CONCLUSION

Our project is barely a humble venture to eliminate the wants to shop for, manage, or maintain servers and different hardware resources. you merely have to be compelled to concentrate on the what and why of readying, rather than the however. Basically, the serverless approach is business-driven, the suggests that by that third parties handle your technical considerations whereas you concentrate on delivering results.

The main aim is to offers variety of benefits over ancient server infrastructure. for several developers, the most drawback with ancient infrastructure is that, it doesn't supply larger quantifiability, a lot of flexibility, and every one resources at a reduced value As several organization tend towards a service bound design, developers can typically wonder if a lot of and a lot of elements of their service may be touched into the cloud. Databases, file storage, and even servers square measure slowly transitioning to the cloud, with servers being run in virtual containers as critical being hosted on a machine. Here the advantages of our application came.

VI. FUTURE ENHANCEMENT

If we tend to remark future improvement we will use AWS cloud services to store the important time generated knowledge and send it to store on cloud. This real time knowledge is extremely helpful for managing significant traffic on our net application. conjointly it might be ready to mechanically and quickly update, patch, fix, or add new options to AN application. it might be ready to mechanically scale in and out servers.

These options ar calculable and rely upon the feedback. These options ar calculable depends upon the necessities of the "Serverless net Application".

The following improvement is also extra to the present modules:-

- Android language will be created bearable.
- MEAN stack will be enforced to create it responsive.
- Speed and nimbleness will be exaggerated supported future out there resources.
- Amazon Alexa skills will be developed.
- Various applications according to different operating System will be developed for make it available for the use of everyone.

Test case id	Purpose	Testing tool	Test Cases	Precondition	Input Test Data	Steps To Be Executed	Expected Result	Actual Result	Pass/Fail
T-AD1	Admin Login	By script in code	Test if admin is able to login	A valid admin account to login	correct username, password	1)admin must be registered 2)enter correct username and password	The admin is logged in successfully	Admin logged in properly	Pass
T-AD2	Admin Registeration	By script in code	Test if admin is able to Register	admin must be registered	admin must selects new user name	1)admin must be registered 2)must have valid username and password	Admin account created	Admin account created successfully	Pass
T-AD3	Create a Book to Category	By script in code	test if admin is able to add a book	must have a category	Admin add a new book to category	1)A category should be there 2)add a new book in category	Book should be updated in Categories list	book added in category	Pass
T-AD4	Delete a Book from Category	By script in code	test if admin is able to delete a book	must have a category	Admin delete a book from category	1)A category should be there 2)delete a book from category	Book should be updated in Categories list	book deleted from category	Pass
T-US5	User Registeration	By script in code	Test if User is able to Register	user must be registered	User must selects new user name	1)user must be registered 2)must have valid username and password	User account created	User account created successfully	Pass
T-US6	User Login	By script in code	Test if user is able to login	A valid user account to login	correct username, password	1)User must be registered 2)enter correct username and password	The user is logged in successfully	User logged in properly	Pass
T-US7	Edit Cart to add a book	By script in code	test if user is able to add a new book in cart	a valid user account	user add a book in cart	1)user must logged in 2)add a book into cart	book should be added	book added successfully	Pass
T-US8	Edit Cart to remove a book	By script in code	test if user is able to remove a book from cart	a valid user account	user remove a book from cart	1)user must logged in 2)remove a book from cart	book should be removed	book removed successfully	Pass
T-US9	Order History	By script in code	User Should be able to watch their order history	must ordered something	user order a book	1)User must be logged in 2)click on order history	ordered book must be visible	order history is vissible to the user	Pass
T-AD10	Block illegal login	Pytest	test if unregistered can login or not	not registered user	incorrect username, password	1)Enter incorrect username, password 2) click submit	user must not be able to log in	Successfully blocked	Pass

ACKNOWLEDGEMENTS

Big thanks to our teachers Ms. Kavita Namdev (Sr. Assistant Professor), Mr. Ritesh Khedekar (Assistant Professor) & Dr. Sanjay Bansal (Head of Department-CSE) for helping us with the project. We also express our deep gratitude to Dr. S. C. Sharma, Director, Acropolis Institute of Technology & Research, Indore for all the help provided for the completion of project.The in-time facilities provided by the department throughout the Bachelors program are also equally acknowledgeable.We would like to convey our thanks to the teaching and non-teaching staff of the Department of Computer Science & Engineering, acme for their invaluable help and support throughout the period of Bachelor's Degree. We are also grateful to all our classmates for their help, encouragement and invaluable suggestions.

REFERENCES

[1] Ben Piper, David Clinton, "AWS Certified Solution Architect Study Guide Associate", Second Edition 2018 released.
[2] Joe Baron, HishamBaz, Tim Bixler, Biff Gaut "AWS Certified Solution Architect official study guide".
[3] Ed, Burnette (July 13, 2010). Hello, Android: Introducing Google's Mobile Development Platform

(3rd ed.). Pragmatic Bookshelf. ISBN 978-1-934356-56-2.
[4] Ableson, Frank; Sen, Robi; King, Chris (January 2011). AWS in Action, Second Edition (2nd ed.). Manning. ISBN 978-1-935182-72-6.
[5] Conder, Shane; Darcey, Lauren (July 24, 2012). AWS Wireless Application Development Volume II: Advanced Topics (3rd ed.).
[6] Murphy, Mark (June 26, 2009). Beginning Android (1st ed.). Apress. ISBN 1-4302-2419-3.
[7] AWS Documentation, Whitepapers & Guides.
[8] Jayendrapatil.com(Blog on AWS resources and services)

