Mulit-Tenant Security Architectures: Design Considerations and Case Studies

SHARMA VAISHALI

I. BACKGROUND OVERVIEW

Cloud infrastructure are increasing rapidly and known as the fast-growing field in the information technology industry. As per the forecast of Gartner's 2019, SaaS based applications will increase from $94.8 billion to $143.7 billion by 2022 (Ihor, 2021). Now, the important question is which one is better Single tenant or multi-tenant and how to design multi-tenant database/system in SaaS? These questions are very important if we want to make sense of tenancy in software services. The present work is an attempt to discuss all the design consideration of multi-tenant architecture system. The study also discusses the important case studies (real-life example studies) that highlight the use of multi-tenant architecture in large educational institutions. In general, a single tenant architecture is a system where the tenant owns an application, and each client uses an isolated database, storage and web server whereas multi-tenant architecture allows clients to run sophisticated apps and large databases immediately in a private and secured environment (Ihor, 2021) at low cost. There are several advantages of multi-tenant architecture such as efficient resource distribution, maintenance free, third party integration and fast scaling. Therefore, designing appropriate multi-tenant architecture as per the client's business needs is very crucial. Some of the models that can be considered are: Database per tenant, Single multi-tenant database, Sharded multi-tenant database and Hybrid sharded multi-tenant database. An in depth discussion of all these design considerations is done in the present work. Several questions were identified and are noted below:

Problem Statement and Research Questions:
Designing appropriate multi-tenant architecture as per the client's business needs is very crucial as it helps in choosing the optimal SaaS architecture for the client's business. The problem is to find the optimal and cost-effective architecture that ensures privacy and security of the client's business. Some of the important questions are:

1. How to make sense of tenancy?
2. Which tenant architecture should be used – Single/Multi?
3. What are the design considerations for Multi-tenant architecture?
4. Can we modify the existing models to provide better solutions?

All these questions will be discussed in the in the present work. Fig. 1 shows the entire workflow of this study.

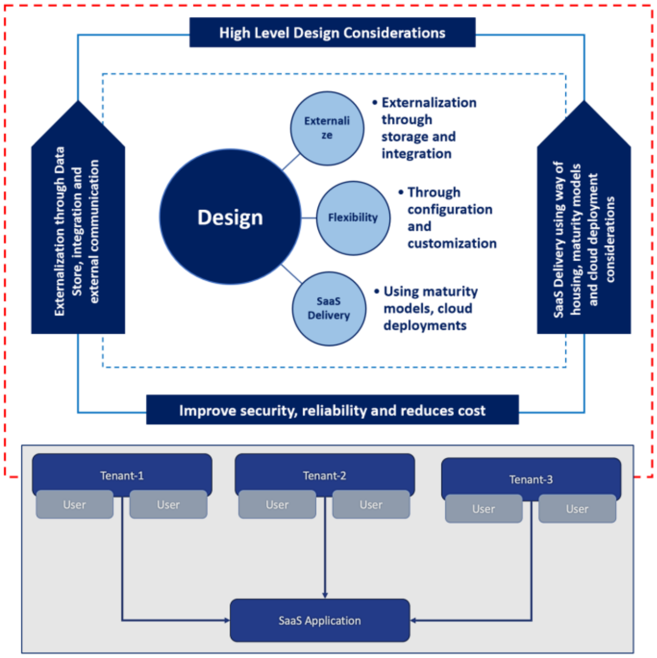
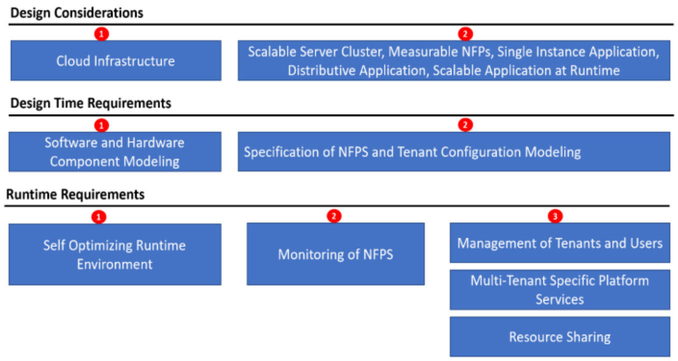

High Level Design Considerations
Externalization through Data Store, integration and external communication
Design
Externalize
• Externalization through storage and integration
Flexibility
• Through configuration and customization
SaaS Delivery
• Using maturity models, cloud deployments
SaaS Delivery using way of housing, maturity models and cloud deployment considerations
Improve security, reliability and reduces cost
Tenant-1
User
User
Tenant-2
User
User
Tenant-3
User
User
SaaS Application

Fig. 1 Workflow of the present study

II. PAST WORK

A multi-tenant cloud architecture is a single cloud instance and infrastructure purpose-built to support multiple customers (Zscaler Cloud Security). Numerous authors discussed various frameworks and designs to develop novel multi-tenant architecture in order to increase the efficiency and security. Table 1 discusses a comprehensive review of all such architecture in detail.

Schroeter et al., 2012 proposed a variable and adaptive SaaS architecture. It is imperative to support the multi-tenant aware architecture at design time as well as at runtime. The authors identified the design time and runtime requirements for variable multi-tenant aware architecture (Fig.2). Design time includes certain applications to be modeled for example an application's software and hardware components and their properties, and tenant configurations. Whereas runtime requirements include evaluation of the modeled information to manage the tenants and their users. The authors also identified the missing concepts for multi-tenancy in cloud control matrix and extended it to structure modeling and variant modeling. The study show how existing architectures for dynamically adaptive applications can be extended for the development and operations of multi-tenant applications.



Design Considerations
1
Cloud Infrastructure
2
Scalable Server Cluster, Measurable NFPs, Single Instance Application, Distributive Application, Scalable Application at Runtime
Design Time Requirements
1
Software and Hardware Component Modeling
2
Specification of NFPS and Tenant Configuration Modeling
Runtime Requirements
1
Self Optimizing Runtime Environment
2
Monitoring of NFPS
3
Management of Tenants and Users
Multi-Tenant Specific Platform Services
Resource Sharing

Fig. 2. Design time and Runtime requirements

Domingo et al., 2010 designed **CLUODIO** cloud computing based metadata powered Multi-Tenant Architecture especially for business information systems. Their system supports the SaaS advanced software delivery approach and designed for managing large datasets. Their two level architecture offers the best flexibility and usability for multi-tenancy database, as it allows managing different users of a system in flexible way. Chong et al., 2006 proposed three approaches to manage the multi-tenant data which includes (i) separating the databases, (ii) shared database, separate schemas, and (iii) Shared database, shared schema. The authors also discussed some business and technical considerations that one should consider before selecting an approach. These considerations include economic considerations, security considerations, tenant considerations, regulatory consideration and skill set consideration. They also discussed the appropriate patterns that can help in planning and building the multi-tenant SaaS application. Pathirage et al., 2011 proposed an innovative multi-tenant architecture especially for business process executions. They showed design and implementation of multi-tenant workflow engine that enables multiple users to run their workflows securely within the same workflow engine instance without the need of modifying the workflow. Povedano-Molina et al., 2013 proposed distributed architecture for resource management and monitoring in cloud named as **DARGOS.** Weber et al., 2019 proposed a scalable platform architecture for blockchain-based systems to ensure data integrity while maintaining data privacy and performance isolation. Wang et al., 2020 explored an effective performance isolation mechanism based on container technology and micro-service architecture in multi-tenant system. This mechanism includes an SLA-oriented multi-tenant and multi-instance hybrid deployment scheme and a tenant performance isolation algorithm for microservice clusters, which improved the cost-efficiency of cloud services and reduced the performance competition among tenants. Czarnecka et al., 2020 proposed a multi-tenant Cloud Computing Architecture that allows the service consumers to share the computing. Levchenko et al., 2021 introduces the functional requirements realization model for implementation and adaptation of multi-tenant SaaS architecture based on reference business process models. Mudigonda et al., 2011 presented a novel multi-tenant architecture, that can exploit inexpensive commodity equipment to scale the network to several thousands of tenants and millions of virtual machines. Kalra et al., 2018 presented a case study to show how microservices can be distributed among multiple tenants to increase resource utilization. Suwarningsih et al., 2020 discussed the process of building a prototype of a multi-tenant system which supports many aspects on the complete multi-tenant, single-tenant database, and single-tenant multi services, respectively.

III. MULTI-TENANT DESIGN CONSIDERATIONS (TAXONOMY STUDY)

Architecting or designing a Multi-tenant application requires a huge effort for handling all complexities from data security to UI display (Swamy, 2014). Numerous frameworks and their usefulness were discussed in the above section. So, it is important to understand how to design such architectures. Normally at high level, three points are considered: Externalize, Flexibility and SaaS Delivery Considerations. They are discussed in the subsequent sections.

STUDY I - EXTERNALIZE DESIGN

Externalizing meaning providing an increased agile operation to a component or feature (Swamy, 2014). It includes three components (shown in Fig. 3):

- Data Store
- Integration
- External Communication

Data Store is used for storing the data (more like a database). During data storage, it is important to consider security of the confidential information of one tenant from other tenant. Data store involves designing of Database (for example database design shown by Vanier, 2019, Howe et al., 2020), defining Data models, Security and compliance of data, Backup and recovery of the data at the low operation cost and Cache. On the other hand, integration involves three components Dependency injection or inversion of control, component and service and Dependency injection and IoC container. Passing a dependency (a service) to a client is known as Dependency injection (Swamy, 2014). Last component is external communication which means transmission of information between a business and another entity in the firm’s external environment (Swamy, 2014).

STUDY II - DESIGN CONSIDERATIONS BASED ON FLEXIBILITY OF THE ARCHITECTURE

Flexibility considers attributes like extended attributes or extra fields, dynamic validation, customization, customization through configuration, customization through security considerations (shown in Fig. 4) Dynamic validation can be enabled using fluent validation API. Customization through configuration is an interesting design attribute. In this system, items are configured based on the process area or unit.

Table 1: Comprehensive Review of Mulit-Tenant Security Architectures (Past Work)

Year	Paper Title	Key findings	Reference
2006	Multi-Tenant Data Architecture	Proposed three approaches to manage multi-tenant data. Also suggested the business and technical considerations that one should follow to identify the best approach.	Chong et al., 2006
2010	CLOUDIO:A cloud computing – oriented Multi-Tenant Architecture for business information systems	Designed CLUODIO cloud computing based metadata powered tqo level Multi-Tenant architecture	Domingo et al., 2010
2011	A Multi-tenant architecture for business process executions	Discussed design and implementation of multi-tenant workflow engine that enables multiple users to run their workflows securely within the same workflow engine instance without the need of modifying the workflow	Pathirage et al., 2011
2011	NetLord: a scalable multi-tenant network architecture for virtualized datacenters.	Presented a novel multi-tenant architecture, that can exploit inexpensive commodity equipment to scale the network to several thousands of tenants and millions of virtual machine	Mudigonda et al., 2011
2012	Towards Modeling a Variable Architecture for Multi-Tenant SaaS-Applications	Proposed a Variable Architecture. Suggested some extensions to CCM structure modeling and Variant Modeling	Schroeter et al., 2012
2013	DARGOS: A highly adaptable and scalable monitoring architecture for multi-tenant clouds	Proposed distributed architecture for resource management and monitoring in cloudS named as DARGOS.	Povedano-Molina et al., 2013
2018	Towards Dynamic Tenant Management for Microservice based Multi-Tenant SaaS Applications	Presented a case study to show how microservices can be distributed among multiple tenants to increase resource utilization	Kalra et al., 2018
2019	A platform architecture for multi-tenant blockchain-based systems	Proposed a scalable platform architecture for blockchain-based systems to ensure data integrity while maintaining data privacy and performance isolation.	Weber et al., 2019
2020	Container based performance isolation for multi-tenant SaaS applications in micro-service architecture	Explored an effective performance isolation mechanism based on container technology and micro-service architecture in multi-tenant system	Wang et al., 2020

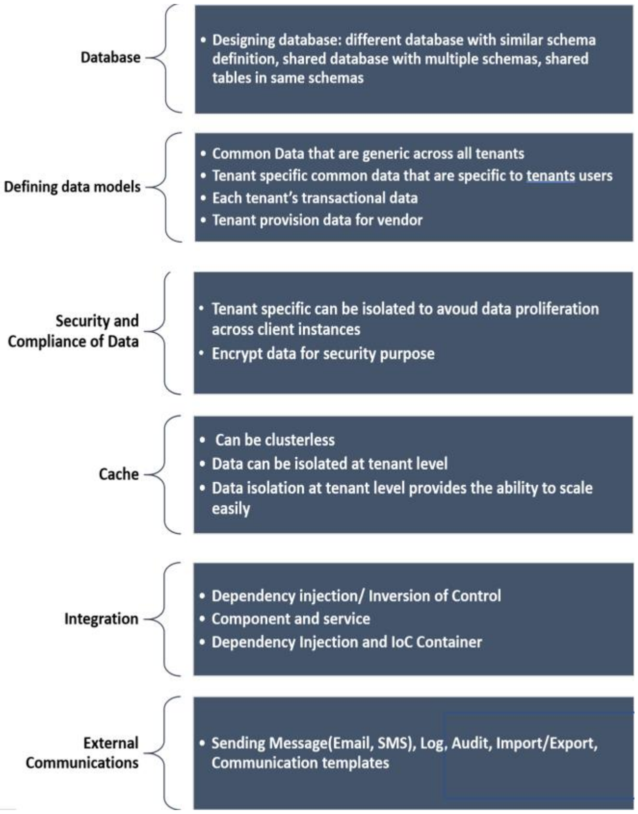

Database
• Designing database: different database with similar schema definition, shared database with multiple schemas, shared tables in same schemas
Defining data models
• Common Data that are generic across all tenants
• Tenant specific common data that are specific to tenants users
• Each tenant's transactional data
• Tenant provision data for vendor
Security and Compliance of Data
• Tenant specific can be isolated to avoud data proliferation across client instances
• Encrypt data for security purpose
Cache
• Can be clusterless
• Data can be isolated at tenant level
• Data isolation at tenant level provides the ability to scale easily
Integration
• Dependency injection/ Inversion of Control
• Component and service
• Dependency Injection and IoC Container
External Communications
• Sending Message(Email, SMS), Log, Audit, Import/Export, Communication templates

Fig. 3 Externize multi-tenant design considerations

For example, if the process area is functional logical unit, then the configurable items will be Functional modules (user management, order management, document management) Feature operation and Feature attributes. For the batch process, configurable items will be parameter and batch process mapping to tenant. Another potential example is Schedular. Frequency, Duration, parameter and Job/batch process will be used as configurable items. User expereince is also very important for which configurable items such as tenant level localization, user level localization and multilingual text for static context could be used.

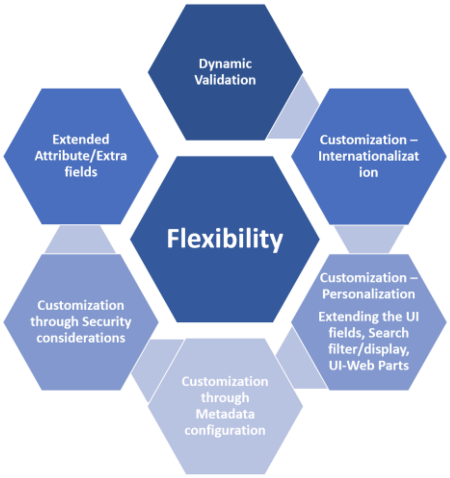

Dynamic Validation
Extended Attribute/Extra fields
Customization – Internationalization
Flexibility
Customization through Security considerations
Customization – Personalization
Extending the UI fields, Search filter/display, UI-Web Parts
Customization through Metadata configuration

Fig. 4 Design considerations based on Flexibility

Some security considerations need to be considered to maintain the flexibility of the Multi-Tenant design architecture. It involves authorization, authentication and federated security and Data Security and Message Security (Swamy, 2014). Lauren et al., 2020 also highlighted security, and efficiency as important parameters that needs to be ensured while designing the multi-tenant architecture.

STUDY III- DESIGN CONSIDERATIONS BASED ON SAAS DELIVERY (DEPLOYMENT AND MATURITY MODELS)

There are two models for SaaS delivery consideration:

- Deployment Models
- Maturity Models

Four approaches are discussed in the Fig. 5 for designing the deployment model. Separate instance and separate database for each tenant could be used as first approach. Another approach is to used same single instance and separate data base for each tenant. Last two approaches involve using shared single instance and single database but different schema or shared single instance and shared database, schema for all tenants (Swamy, 2014). On the other hand, maturity models are designed based on the customization type. For example, if it is individually customized, configural maturity model could be used that provides single instance serving each and every customer.

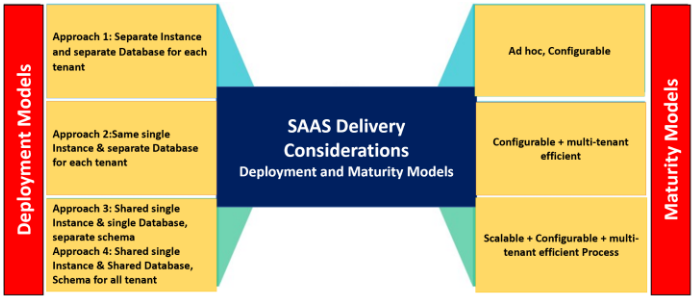

Deployment Models
Approach 1: Separate Instance and separate Database for each tenant
Approach 2:Same single Instance & separate Database for each tenant
Approach 3: Shared single Instance & single Database, separate schema
Approach 4: Shared single Instance & Shared Database, Schema for all tenant
SAAS Delivery Considerations
Deployment and Maturity Models
Ad hoc, Configurable
Configurable + multi-tenant efficient
Scalable + Configurable + multi-tenant efficient Process
Maturity Models

Fig. 5. SaaS Delivery Considerations: Deployment/Maturity

IV. CASE STUDY I

MULTI-TENANT ARCHITECTURE DESIGN FOR EDUCATIONAL LARGE INSTITUTIONS

Lauren et al., 2020 designed a multi-tenant architecture for large institutions wherein they discussed the essential design principles for reducing the cost and increasing the efficiency and security. They suggested to create separate tenants to improve the educational environment. Following separation and considerations were proposed in the study of Lauren et al., 2020

- Administrative separation
- Resource separation
- Configuration separation
- Administrative considerations
- Resource considerations
- Configuration considerations

They proposed the regional approach when deploying multiple tenants, because this approach can help in minimizing the number of users moving across tenants. It also helps in optimal collaboration within each region. **Azure AD B2B collaboration** enables users to use one set of credentials to sign into multiple tenants and it is very beneficial for large educational institutions as it provides:

- Centralized administration team managing multiple tenants (as shown in Fig. 6)
- Teacher collaboration in the regions
- Onboarding parents and guardians with their credentials
- External partnerships

Several other methods were also discussed by Lauren et al. (2020) which include using apps across multiple tenants, per-tenant administration, delegation of admin roles with a tenant, cross tenant management and managing objects at scale using **Microsoft Graph and Azure AD PowerShell**. They all could be helpful in designing multi-tenant architecture for large educational institutions.

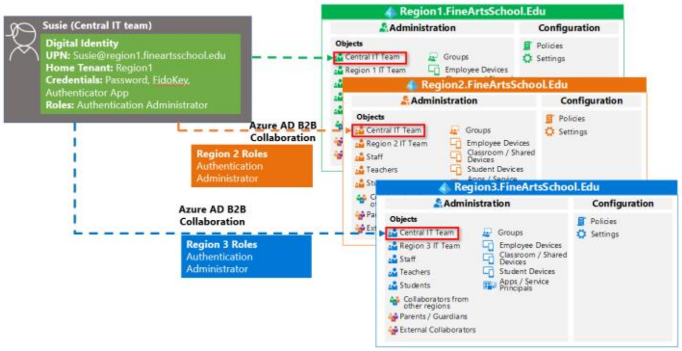


Fig. 6 Centralized administration of multiple tenants (Source: Lauren et al., 2020)

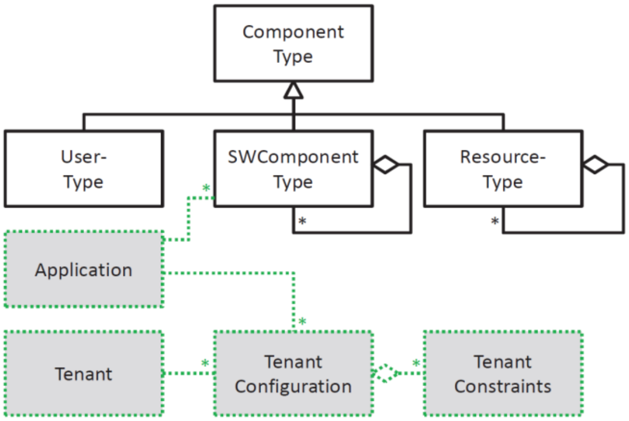


Fig. 7 Extensions to the cloud control matrix structure modeling highlighted with dashed lines (Source: Schroeter et al., 2012)

V. Case Study II

Variable Architecture Design for Multi-Tenant SaaS Applications

Schroeter et al., 2012 proposed a variable and adaptive multi-tenant SaaS architecture. It is imperative to support the multi-tenant aware architecture at design time as well as at runtime and the authors identified all the design time and runtime requirements (as shown in Fig. 2) for variable multi-tenant aware architecture. Schroeter et al., 2012 also discussed how existing architecture for dynamically adaptive applications can be extended for the development and operations of multi-tenant applications. The authors also identified the missing concepts for multi-tenancy in cloud control matrix and extended it to structure modeling and variant modeling (as shown in Fig. 7)

References

[1] Chong, F., Carraro, G., & Wolter, R. (2006). Multi-tenant data architecture. *MSDN Library, Microsoft Corporation*, 14-30.

[2] Czarnecka, P. (2020). The Multi-Tenant Cloud Computing Architecture Allows the Service Consumers to Share the Computing. *Tennessee Research International of Social Sciences*, *2*(1), 1-21.

[3] Domingo, E. J., Nino, J. T., Lemos, A. L., Lemos, M. L., Palacios, R. C., & Berbís, J. M. G. (2010, July). Cloudio: A cloud computing-oriented multi-tenant architecture for business information systems. In *2010 IEEE 3rd International Conference on Cloud Computing* (pp. 532-533). IEEE.

[4] Howe, B. (2020, January 5). *Multi-tenant Application Database Design, Medium.* Retrieved from http://blakebhowe.medium.com/multi-tenant-application-database-design-e4c2d161f3dd

[5] Ihor, F. (2021, February 4). Relevant Software | Software Development Outsourcing Company. Multi-Tenant Architecture: Designing a SaaS Application. Retrieved from http://relevant.software/blog/multi-tenant-architecture/

[6] Kalra, S., & Prabhakar, T. V. (2018, February). Towards dynamic tenant management for microservice based multi-tenant saas applications. In *Proceedings of the 11th Innovations in Software Engineering Conference* (pp. 1-5).

[7] Lauren et al. (2020, August 24). Developer tools, technical documentation and coding examples | Microsoft Docs. *Multi-tenant architecture for large institutions - M365 Education | Microsoft Docs.* Retrieved from http://docs.microsoft.com/en-us/microsoft-365/education/deploy/design-multi-tenant-architecture

[8] Levchenko, A., & Taratukhin, V. (2021, January). Reference Business Processes-Based Method for Multi-Tenant SaaS Architecture Deployment and Adaptation. In *2021 28th Conference of Open Innovations Association (FRUCT)* (pp. 264-270). IEEE.

[9] Mudigonda, J., Yalagandula, P., Mogul, J., Stiekes, B., & Pouffary, Y. (2011). NetLord: a scalable multi-tenant network architecture for virtualized datacenters. *ACM SIGCOMM Computer Communication Review*, *41*(4), 62-73.

[10] Pathirage, M., Perera, S., Kumara, I., & Weerawarana, S. (2011, July). A multi-tenant architecture for business process executions. In *2011 IEEE International Conference on Web Services* (pp. 121-128). IEEE.

[11] Povedano-Molina, J., Lopez-Vega, J. M., Lopez-Soler, J. M., Corradi, A., & Foschini, L. (2013). DARGOS: A highly adaptable and scalable monitoring architecture for multi-tenant Clouds. *Future Generation Computer Systems*, *29*(8), 2041-2056.

[12] Schroeter, J., Cech, S., Götz, S., Wilke, C., & Aßmann, U. (2012, January). Towards modeling a variable architecture for multi-tenant SaaS-applications. In *Proceedings of the sixth international workshop on variability modeling of software-intensive systems* (pp. 111-120).

[13] Suwarningsih, W., Heryana, A., Pardede, H. F., Adhi, P., Zilvan, V., Rozie, A. F., ... & Arisal, A. (2020, November). Qu-Antri: Multi-tenant Architecture for Bandung Public Services to Reduce the Crowd. In *2020*

International Conference on Radar, Antenna, Microwave, Electronics, and Telecommunications (ICRAMET) (pp. 135-140). IEEE.

[14] Swamy, A. (2014, October 06). CodeProject - For those who code. *Multi-Tenancy Design Consideration - CodeProject.* Retrieved from http://www.codeproject.com/articles/809851/multi-tenancy-design-consideration

[15] Vanier , E. (2019). . *Multi-tenant Database Architecture: All you need to know about it - MySQL Database Performance | performed without giving any access to your production data.* Retrieved from http://ericvanier.com/news/multi-tenant-database-architecture-all-you-need-to-know-about-it/

[16] Wang, Y., Sun, Y., Lin, Z., & Min, J. (2020, April). Container-Based Performance Isolation for Multi-Tenant SaaS Applications in Micro-Service Architecture. In *Journal of Physics: Conference Series* (Vol. 1486, No. 5, p. 052032). IOP Publishing.

[17] Weber, I., Lu, Q., Tran, A. B., Deshmukh, A., Gorski, M., & Strazds, M. (2019, March). A platform architecture for multi-tenant blockchain-based systems. In *2019 IEEE International Conference on Software Architecture (ICSA)* (pp. 101-110). IEEE.

[18] Zscaler (n.d.). Secure Digital Transformation | Zscaler Cloud Security. *What is a Multitenant Cloud Architecture ? | Zscaler*. Retrieved from http://www.zscaler.com/resources/security-terms-glossary/what-is-multitenant-cloud-architecture