Budapest University of Technology and Economics
Faculty of Electrical Engineering and Informatics
Department of Automation and Applied Informatics

Development of a template-driven website generator SaaS platform

Master's Thesis

Author
Péter Hanich

Advisor
Gábor Knyihár

December 8, 2025

Contents

HALLGATÓI NYILATKOZAT

Alulírott *Hanich Péter*, szigorló hallgató kijelentem, hogy ezt a diplomatervet meg nem engedett segítség nélkül, saját magam készítettem, csak a megadott forrásokat (szakirodalom, eszközök stb.) használtam fel. Minden olyan részt, melyet szó szerint, vagy azonos értelemben, de átfogalmazva más forrásból átvettem, egyértelműen, a forrás megadásával megjelöltem.

Hozzájárulok, hogy a jelen munkám alapadatait (szerző(k), cím, angol és magyar nyelvű tartalmi kivonat, készítés éve, konzulens(ek) neve) a BME VIK nyilvánosan hozzáférhető elektronikus formában, a munka teljes szövegét pedig az egyetem belső hálózatán keresztül (vagy autentikált felhasználók számára) közzétegye. Kijelentem, hogy a benyújtott munka és annak elektronikus verziója megegyezik. Dékáni engedéllyel titkosított diplomatervek esetén a dolgozat szövege csak 3 év eltelte után válik hozzáférhetővé.

Budapest, 2025. december 8.

Hanich Péter
hallgató

Abstract

Chapter 1

Introduction

Chapter 2

Literature review

2.1 Software as a Service

In order to develop a software as a service (SaaS) application suite we first have to explore what a SaaS entails. To do this an overview and understanding of cloud computing and its related terminologies is necessary. ISO defines cloud computing as a "paradigm for enabling network access to a scalable and elastic pool of shareable physical or virtual resources with self-service provisioning and administration on-demand. [1]" The NIST (National Institute of Standards and Technology) defines five essential characteristics, four deployment models and three service models for the cloud. [2]

The essential characteristics are the following:

- On demand self-service: resources can be provisioned without additional human assistance.
- Broad network access: users can access capabilities through standardized methods over the network.
- Resource pooling: the provider manages their cloud resources in such a way that it can serve multiple customers flexibly.
- Rapid elasticity: capabilities are scalable and reassignable on demand.
- Measured service: resource usage is monitored, controlled and automatically optimized.

The aforementioned deployment models are:

- Private cloud: the cloud is used by a single organizational entity
- Community cloud: the infrastructure is used by a community composed of multiple organizations or
- Public cloud: the cloud is open for use to the public.
- Hybrid cloud: the cloud is some mix of the models mentioned before.

All the above deployment models may be either on or off premise and additionally owned or managed by the organization(s) using them, third parties or some combination of the former.

Most importantly for the topic the three service models:

- Infrastructure as a Service (IaaS): the provided resources are fundamental to computing, such as network, storage or processing power.
- Platform as a Service (PaaS): the provided resources allow the customer to deploy applications to the cloud providers platform without configuring the underlying computing infrastructure.
- Software as a Service (SaaS): "The capability provided to the consumer is to use the provider's applications running on a cloud infrastructure. The applications are accessible from various client devices through either a thin client interface, such as a web browser (e.g., web-based email), or a program interface. The consumer does not manage or control the underlying cloud infrastructure including network, servers, operating systems, storage, or even individual application capabilities, with the possible exception of limited user-specific application configuration settings." [2]

There are many cloud providers available for use most notably Amazon Web Services (AWS), Microsoft's Azure and Google Cloud Platform (GCP), which boast numerous amount services belonging to the models described previously.

2.1.1 Aspects and challenges of SaaS

After establishing a definition of SaaS and its related terminologies, lets dive into the different aspects and challenges of developing, maintaining and operating one.

"Software as a service (SaaS) is a software delivery model in which third-party providers provide software as a service rather than as a product over the Internet for multiple users." [3] SaaS applications hence are also web applications and built with web technologies and frameworks. However not all web applications are SaaS due to the aforementioned delivery model. In a way SaaS is a specific subset of web applications.

There is not a fixed defined set of characteristics that a SaaS has to meet in order to be considered one, however there are some commonly recurring ones that most take into account, such as multi-tenancy, customization and scalability. [4]

Most SaaS applications support some form of multi tenancy. Multi tenancy is the ability to share the same (and single) hosted instance of the application between multiple tenants, instead of deploying the application separately for each tenant. A tenant is a user or group of users (e.g. an organization) with the same share of the service. [5] Commonly people refer to multi-tenancy as the ability to have multiple organizations use the service with their own share and users, however as we can see this is not a strict requirement. There are many ways to achieve multi-tenancy with different amounts of complexity in concern areas like architecture, scaling, and security. The methods usually differ by how separated each tenants share is on a database, data model and infrastructure level. [6]

This goes hand-in hand with the next common characteristic which is customization. SaaS applications must support a level of customization to meet all tenants needs. This allows tenants to adapt the service more seamlessly into their workflows, processes and requirements. The ability to change aspects also reduces repetitive tasks between users of the same tenants. Customization can range from interface level differences to full on tailoring of the tenants share of the application down to the platform or even infrastructure level. Each added aspect of personalization however increases complexity, due to which a balance must be struck between customization and standardization. [7]

Scalability is the ability of the service to handle increasing workloads and usage needs. Generally there are two methods of scaling, vertical scaling (also known as scale-up) and horizontal scaling (also known as scale-out). Vertical scaling involves giving more resources to the machine running the service, while horizontal scaling means distributing the application on multiple machines. In a cloud environment a scale-out solution is relied on more by default due to the available amount of resources and also due to the fact that increasing a machines resources is only feasible up to a level. However, most complex production system usually combine the two forms in some form. Not only that SaaS platforms commonly employ either a two level or a generalized K-level scalability structure where each level is a separate dimension of the service capable of scaling independently. Scaling solutions are also made tenant aware so that each tenant gets scaled for their respective needs. [8]

Of course plenty more aspects could be reviewed and explored regarding SaaS applications, however the previously mentioned concerns and areas are sufficient to get a general understanding required for the topic.

2.2 Laravel

2.3 React

2.4 Kubernetes

Chapter 3

Development

3.1 Architecture

3.2 Backend

3.3 Frontend

3.4 Proxy application

Chapter 4

Testing

4.1 Backend testing

4.2 Frontend testing

4.3 Usage example

Chapter 5

Production considerations

Chapter 6

Summary

Acknowledgements

Bibliography

[1] ISO/IEC22123-1. Information technology — Cloud computing Part 1: Vocabulary. Standard, International Organization for Standardization, Geneva, CH, February 2023.

[2] Peter Mell, Tim Grance, et al. The nist definition of cloud computing. 2011.

[3] Saiqa Aleem, Rabia Batool, Faheem Ahmed, Asad Khatak, and Raja Muhammad Ubaid Ullah. Architecture guidelines for saas development process. In *Proceedings of the 2017 International Conference on Cloud and Big Data Computing*, pages 94–99, 2017.

[4] Javier Espadas, David Concha, and Arturo Molina. Application development over software-as-a-service platforms. In *2008 the third international conference on software engineering advances*, pages 97–104. IEEE, 2008.

[5] Rouven Krebs, Christof Momm, and Samuel Kounev. Architectural concerns in multi-tenant saas applications. *Closer*, 12:426–431, 2012.

[6] WeiTek Tsai, XiaoYing Bai, and Yu Huang. Software-as-a-service (saas): perspectives and challenges. *Science China Information Sciences*, 57(5):1–15, Mar 2014. DOI: `https://doi.org/10.1007/s11432-013-5050-z`. URL `https://link.springer.com/article/10.1007/s11432-013-5050-z`.

[7] Farhan Aslam. The benefits and challenges of customization within saas cloud solutions. *American Journal of Data, Information and Knowledge Management*, 4(1): 14–22, 2023.

[8] Wei-Tek Tsai, Yu Huang, Xiaoying Bai, and Jerry Gao. Scalable architectures for saas. 04 2012. DOI: `10.1109/ISORCW.2012.44`.

Appendix