VIETNAM NATIONAL UNIVERSITY OF HOCHIMINH CITY

THE INTERNATIONAL UNIVERSITY

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING



MERN STACK WITH GRAPHQL FOR SHARE ACCOMODATION APPLICATION

**Advisor: MSc. Dao Tran Hoang Chau**

**Student name: Le Nguyen Nhat Minh**

A thesis submitted to the School of Computer Science and Engineering

in partial fulfillment of the requirements for the degree of   
Bachelor of Information Technology

Ho Chi Minh City, Vietnam

2021

MERN STACK WITH GRAPHQL FOR SHARE ACCOMODATION WEB APPLICATION

APPROVED BY APPROVED BY COMMITTEE

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Dao Tran Hoang Chau, MSc**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**THESIS COMMITTEE**

# ACKNOWLEGMENTS

During the period of challenging time with Thesis. With a sincere heart, I would like to say thanks to my supervisor MSc. Dao Tran Hoang Chau, who directly guided me and to be always willing answered my questions, instructed me handle with complex problems as well as support me various precious ideas in the case of extending my Thesis.

In the process of studying, I’m also grateful Dr. Tran Thanh Tung for supporting technical implementation and suggestions. Furthermore, I really appreciate teachers in Computer Science and Engineering Department for enthusiastically conveying deep knowledge, advices to student.

Finally, I would like to thank my friends at International University for encouragement and help me in the process of doing thesis project.

# TABLE OF CONTENTS

ACKNOWLEGMENTS

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1 INTRODUCTION

1.1 Motivation

1.2 Problem Statement

1.3 Scope and Objectives

1.4 Assumption

CHAPTER 2 TECHNOLOGIES

2.1 Background of query standards

2.1.1 REST

2.1.2 GraphQL

2.2 GraphQL Architecture

2.2.1 Simple architecture

2.2.2 Existing systems

2.2.3 Hybrid systems

2.3 The contrast of GraphQL and REST Architecture

2.4 Overview of MERN stack

2.4.1 MongoDB

2.4.2 Express

2.4.3 ReactJS

2.4.4.Node.js

2.5 Other Technologies

2.5.1 Apollo

2.5.1.1 Apollo Server

2.5.1.2 Apollo Client

2.5.2 JSON Web Tokens

Chapter 3 METHODOLOGY

3.1 Functional Requirement

Chapter 4 SYSTEM DESIGN

4.1 Backend (Web Server)

4.2 Frontend (Web UI)

# LIST OF ABBREVIATIONSALS

REST Representational State Transfer

API Application Programming Interface

URL Uniform Resource Locator

HTTP Hypertext Transfer Protocol

JSON JavaScript Object Notation

IDE Integrated Development Environment

SDL Schema Definition Language

CRUD Create, Read, Update, and Delete

UI User Interface

SPA Single-page Application

# ABSTRACT

In the modern world, office, factory or school often focus on the industry zone or in the big cities. However, not everyone also lives in these areas. Most of them come from the rural areas or very far away from the workplace. In addition, some of them are just alone, their budget may be also limit for spending on renting an apartment, house or even room. Furthermore, migration from other places, they have quite troubles in finding a place for livings. Base on that demand, share accommodation website has been released to adapt these requirements.

However, there are few available websites in Vietnam focus on that problems. Almost, these websites mainly provide rental apartments, houses or rooms and lack of share accommodation. Moreover, those websites are now built rudimentarily and have some limitations in common case such as bottle-neck, poor interactions or overloading due to some weak aspects of REST. For that reasons, GraphQL is applied to this thesis to solve these issues, boost performance and make the interaction between server and client become less complicated. Moreover, MERN stack is powerful and effective to work with GraphQL.

# CHAPTER 1 INTRODUCTION

* 1. Motivation

The Share Accommodation is simple way to explain that house owners share their home with other renters to reduce rental fees. In the share accommodation application, users are permitted to find renters who are registered to search a house or an apartment. Also, renters can find available houses or apartments which is posted for sharing.

In addition, GraphQL is applied to this application to resolve RESTful API limitation and boost performance higher.

* 1. Problem Statement

In the present time, there are very few websites focus on the house share or flatmate finder. However, these websites were almost built very basic and show less information or even just a static web. Moreover, they have troubles when receive tons of submitting request at the same time lead to be overloaded or died. Although microservices was born to resolve these problems, this type of project is not big enough to use microservices in order to scale server. That is the reason why GraphQL is used to point out the limit aspect of REST.

* 1. Scope and Objectives

The problem of RESTful API is that they load too much data without necessary purposes. Therefore, I use new technologies to tackle that weakness of REST and be able to develop bigger in the future through different levels such as:

* Web(client), Server(backend) and GraphQL(query language).
* Web: ReactJS is a strong javascript library for creating interactive UIs. It functionally updates and change data with rendering right components. Besides, friendly view design make code cleaner.
* Server: NodeJS work as non-blocking I/O to develop scalable network application and its ability can handle a huge amount of concurrent operations.
* GraphQL: A query language for APIs was developed by Facebook permits clients the power to request what data they need and nothing more.
  1. Assumption

Due to the barrier of time and complex of the issues, the thesis has not been absolute done. It need to be improvement and development in the future. Some of features in the list of function is not ensured to complete. However, I would like to list several assumptions that may finish later:

* Landlord: a specified page for house owner to manage their member. It has still not built now. In final product, it might be added to make the app more interactive.
* Dashboard: there are a lot chart in order to analyze user behavior built. However, it seems to be need more chart to visualize data in details

# CHAPTER 2 TECHNOLOGIES

This chapter will cover basic background of technologies are used in the thesis project.

**2.1 Background of query standards**

In the technology world, a wide variety of query standards that apply to various environments such as SQL (Structured Query Language), Graph Query Language, REST (Representational State Transfer) and more flexible others.

2.1.1 REST

Representational State Transfer or REST which was first introduced in 2000 by R.F (Roy Fielding). REST become popular in the next year due to its architectural concept which support to design distributed systems. REST is a much lighter and high performance compared with SOAP (commonly used at that time). Noticeably, REST was born to solve problems about bandwidths that SOAP could not handle.



Figure 1 SOAP vs. REST: The key differences (By Upwork Staff, 2017)

2.1.2 GraphQL

In GraphQL, it was released in the recently by Facebook in 2015 and officially open source in 2018. They wanted to find a solution to deal with the increasing of mobile applications. Because of the weakness of mobile devices, they are actually hard for loading a heavy website as Facebook. Fetching multiple queries to do a task being really wasted. Consequently, they developed GraphQL to wrap all data needed into a single query. Thus, performance is improved significantly.

**2.2 GraphQL Architecture**

As mentioned above, GraphQL is a query language for API as well as a server-side runtime in order to execute queries based on a type system defined by programmer. This technology also decreases complicated logics to restructure the data after executing multiple fetches.

2.2.1 Standard Architecture

The simple architecture between client and a server that implements the GraphQL. As a query arrives., the server reads the query’s payload and get the essential information from database

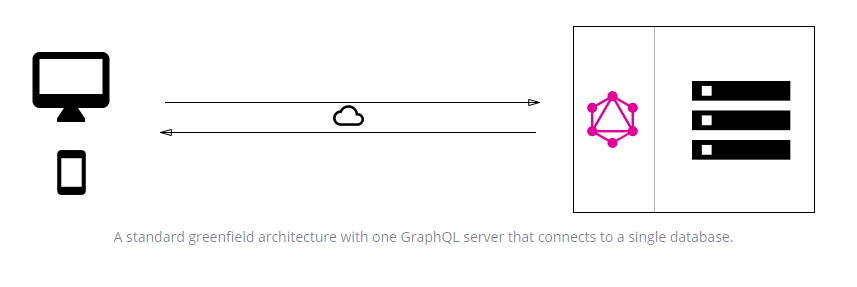


Figure 2 A common client-server architecture with GraphQL

(howtographql, 2021)

Then, it will return the response object which is constructed as the response format to the client. Noticeably, GraphQL runs as transport-layer agnostic means that it is able to be used with another network protocol such as TCP, WebSockets, etc…. Moreover, GraphQL was designed flexibly to handle with any database such as SQL (PostgreSQL), NoSQL (MongoDB).

2.2.2 In Case of Existing Systems

One of powerful GraphQL is the integration of multiple existing systems. That’s quite great for many companies with legacy infrastructures and APIs that have developed for long years and require maintenance now. However, it is hard for these companies to connect multiple systems. In that case, GraphQL is an amazing choice to unify these existing systems without considering their complexity.

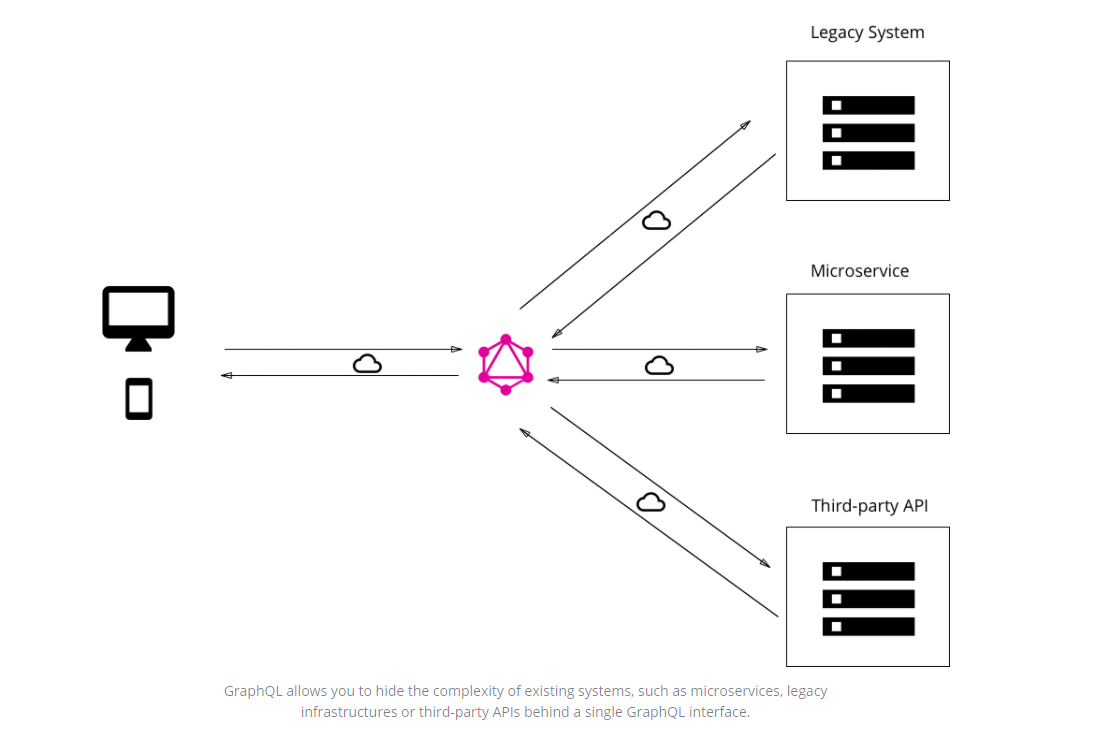


Figure 3 GraphQL as a gateway for existing systems (howtographql, 2021)

As the same previous architecture, the GraphQL server just does its work that receives query from client, then wraps data fetching from existing systems in GraphQL response format and response to client.

2.2.3 Hybrid Architecture

In this architecture, the GraphQL server plays a centre gate way as primary end point in the centre of the system.

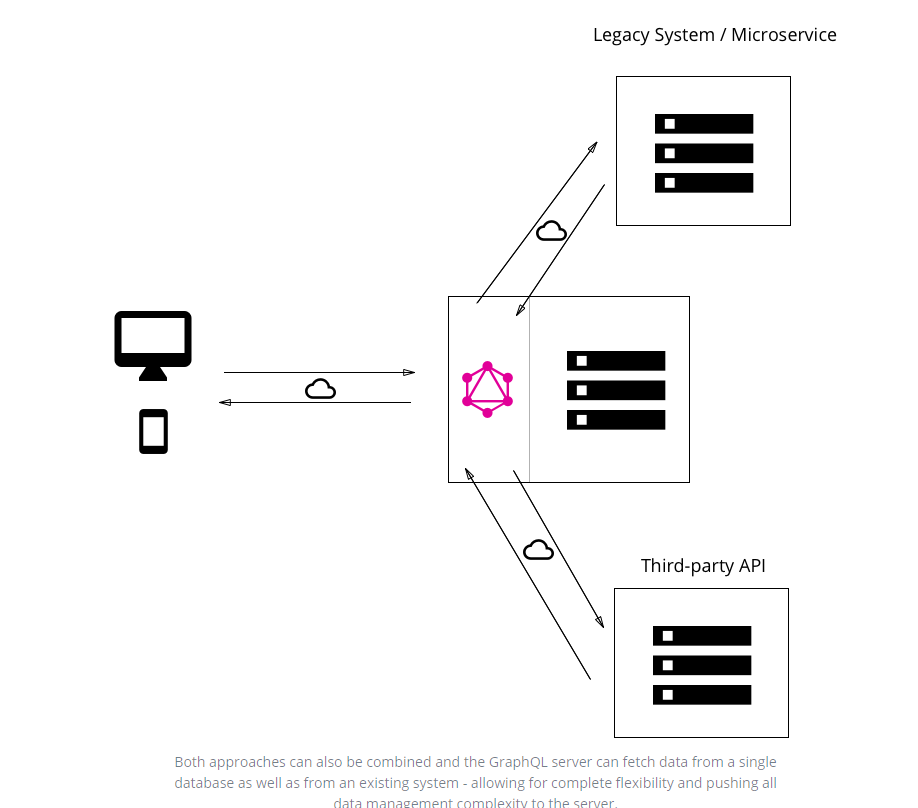


Figure 4 Hybrid GraphQL architecture (howtographql, 2021)

This API bridge directly accesses to database as a basic first version. A query is received by the server, then it response either to database connected or some of the system integration after resolve that query.

2.3 GraphQL Vs. REST architecture

The number of available endpoints is the primary differences between those methods. While GraphQL has only single endpoint to combine every request, REST tries to gather data by accessing multiple spots.

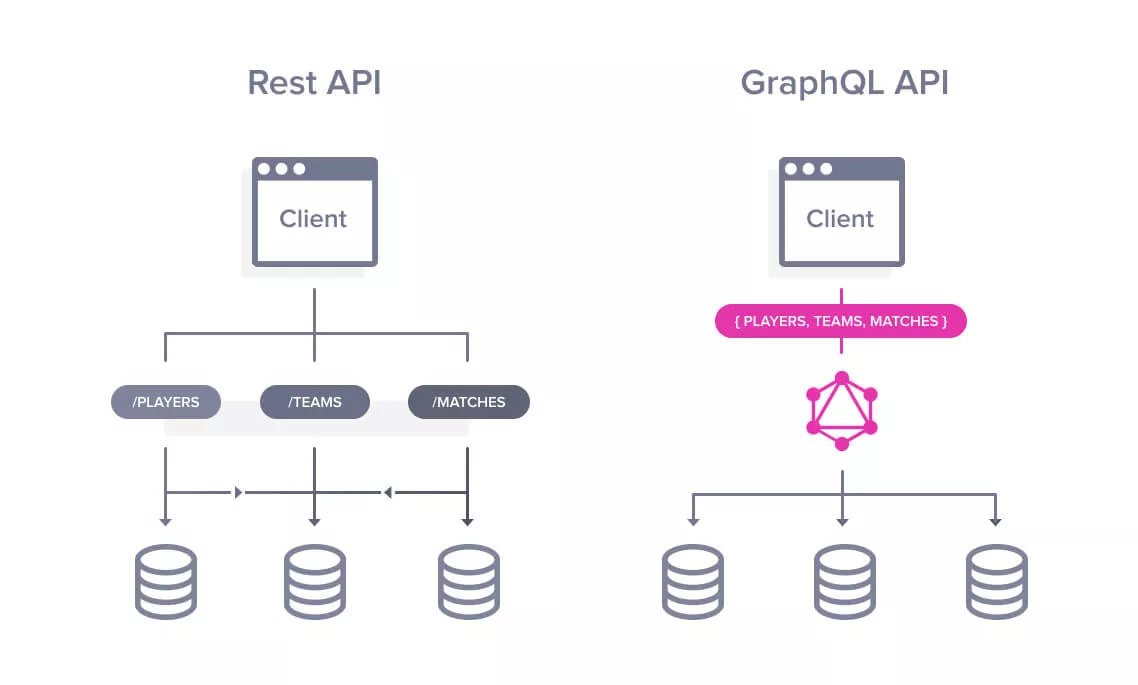


Figure 5 diagrams illustrate how REST and GraphQL work (devopedia, 2021)

In the case of the REST, it must be right endpoints when up to client or query. This causes a situation that backend changes something can lead to mismatch with the frontend if programmer forget to modify the frontend.

2.4 Overview of MERN stack

2.4.1 MongoDB

MongoDB is a scalable NoSQL database, and open source. In contrast with current database, table, record in SQL is replaced by collection and document in MongoDB. A document is written by field and value structure. Data is displayed as a shape of binary JSON or BSON.

In the fact that, there are too many options database to select one. But, MongoDB database is chosen for several reasons:

* Pros of MongoDB
  + - Dynamic and expressive query language that allows to filter or sort by any field,
    - No more concatenating strings to dynamically generate SQL queries.
    - No concatenate strings to generate dynamical SQL queries.
    - No schema- Adding new field without alter table
    - Relationships management become simplified
    - Strongly generalize schema
    - Database and schema can be mapped by using ORM
* Cons of using MongoDB
  + - No enforcement for types and database structure. Listing data can occur some GraphQL errors
    - Resolvers must check external source of the data be right with the defined schema to avoid breaking GraphQL API.
    - Adding new required fields, developers need to provide the way to return default values or new value.

Due to the advantages overweigh the disadvantages, MongoDB is select instead of other databases.

2.4.2 Express

In web application, express is a flexible and minimal framework of Node.js to develop web server. It was designed with a large of HTTP utility methods and middleware that APIs could be customized by programmers.



Figure 6: Express Framework

Due to some different architecture with REST, GraphQL developer is provided other express framework version called express-graphql. It support GraphQL HTTP server connect styled middleware.

2.4.3 ReactJS

React as a JavaScript library developed by Facebook for building UIs. It cannot be denied the popular of React in many website applications due to its SPA feature. Based on data changes, just the right components will be updated and render effectively.

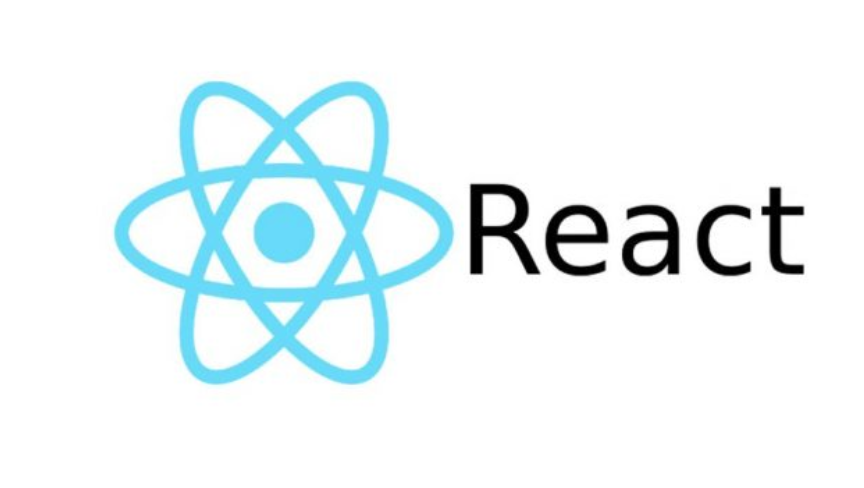


Figure 7 React Icon

Moreover, ReactJS code is quite clean, it normally predictable and easier for us to debug. It abstracts away the DOM, simplifies programming model and better performance.

2.4.4 Node.js

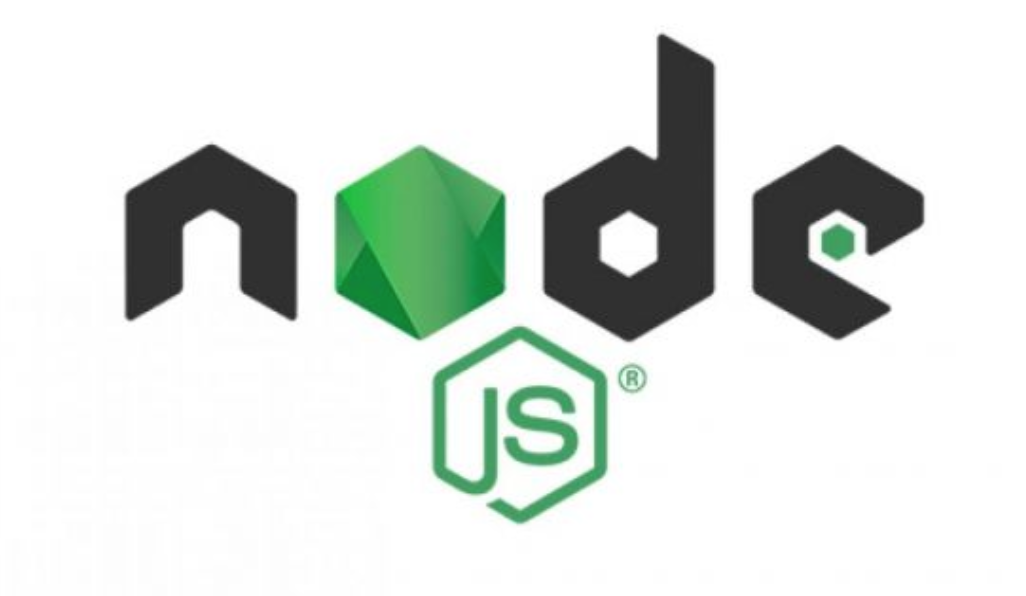


Figure 8 Node.js Icon

It is no double about Node.js, a JavaScript-based platform, built on Google V8 JavaScript. It used to develop non-blocking I/O API and asynchronous web application like SPA, video streaming, etc…Thanks to I/O model, real time, asynchronous, faster, scalable are the noticeable points of Node.js.

2.5 Other Technologies

2.5.1 Apollo

Apollo is a platform which is used to build, query and control a data graph. It is a centre gate way to permit applications communicate between clients and backend services. Additionally, It uses GraphQL to build and enforce the data flow pattern.

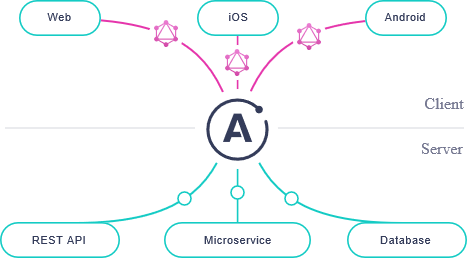


Figure 9 Workflow Apollo

It is noticed that GraphQL can be actually built without Apollo platform, there are many available libraries for GraphQL in the npmjs package but those libraries are not flexible, need to be customized for specified task and make code less clean. Moreover, some GraphQL platform are same as Apollo like Relay, urql,… However, Apollo is not only powerful, flexible but also easy-to-understand.

2.5.1.1 Apollo Server

Apollo Server is an open source GraphQL server by using JavaScript. It is one of GraphQL server libraries being fast, simple and built on top of Express.js This service can connect to data source from backend and work with data. Besides, it can be compatible with other GraphQL clients. Apollo Server are possibly used as an independent GraphQL server or add-on to existing application Node.js middleware (express, Fastify). This provides:

Straightforward setup: fast fetching data

Incremental adoption: Adding essential features

Universal compatibility: with other API.

Production readiness: shipping features quickly

2.5.1.2 Apollo Client

Apollo Client is a JavaScript library concerns state management which enables to control both local and remote data with GraphQL. This service is production-ready and can fetch, cache, modify data but UI still keep auto update. The core Apollo client was built to integrate with React. Especially, its syntax based on React Hooks.

2.5.2 JSON Web Token (JWT)

JWT, an open standard (RFC 7519), describes the way of information may be transmitted as a compact JSON object. It commonly used for authorization to securely protect API. In this project, we require a valid access token to be sent in the Authorization header of each request. The structure of JWT consists of three distinct parts divided by dots(.):

Header: containing token type and use the Algorithm to sign the token

Payload: having claims about an entity and additional data. These statement either have predefined claims (registered) or can be defined by JWT user (public/private claims).

Signature: to verify nothing changes during the transmission of token by hashing header, payload and a secret key declared.

The output is three Base64-URL strings and use dot(.) to concat these strings.

A JWT will look like this:

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9*.eyJ1c2VySWQiOiI1ZmQwYWNmMDdlNT.*L6mXymNKGMP9xsE3WPvYKP9BClXDdPsFtHIQfkvCoQU

All of the information inside can be decoded again.

# CHAPTER 3 METHODOLOGY

3.1 Functional Requirement