Опис на архитектурните стилови

Проектот „КОВИД-Центри“ претставува систем кој ќе овозможува низа на функции кои ќе можат да бидат извршувани од страна на системот но и исто така ќе им нуди на корисниците пријатно корисничко искуство. Со цел да им овозможиме на корисниците целосна функционалност, пребарување и навигирање на КОВИД-центрите, потребно е да имплементираме систем кој ќе вклучува различна структурална организација .

Извршната архитектура претставува архитектура која се фокусира на runtime структурата на системот и ги опфаќа хардверските елементи, подсистемите, процесите и нишките.
Во нашиот проект користиме клиент-сервер архитектура каде што клиентот испраќа барање до серверот за одредена активност, серверот го обработува тоа барање и го праќа назад кон клиентот. Во овој дел спаѓа и филтрирањето на податоците од мапите кои подоцна се стават во базата на податоци.

Имплементациската архитектура се однесува на тоа како системот е изграден. Компонентите може да бидат апликациски и инфраструктурни. Најчесто инфраструктурните компоненти служат како контејнер за останатите.
Функционира на тој начин така што веб серверот комуницира со инфраструктурните компоненти кои пак ги земаат информациите од базата на податоци.
Нашата апликација е од типот RESTful со thin clients каде што целата логика на е на серверска страна.

Концептуалната архитектура се фокусира на одговорности кои се однесуваат на ниво на домен. Иницијални чекори на целиот процес се идентификување на клучни концепти и сместување на истите во категории.
Component responsabilities:
1. Кориснички интерфејс
-прикажување на места
-прикажување на рута
-прикажување на листа со КОВИД-центри
2. Управувач со датотеки
-проверка/додавање на податоци со базата на податоци

Data-flow архитектурата ќе ни овозможи превземање на податоците и мапата, а потоа и соодветно нејзино филтрирање со користење на *pipes&filters* односно екстрахирање на потребните податоци со цел нивно понатамошно користење.

Податоците ќе ги зачуваме во соодветна база на податоци т.е ќе користиме *data-centered* архитектура каде што секој корисник испраќа барање до серверот а серверот му одговара соодветно на тоа барање со даден одговор (*repository*).

Нашиот систем ќе се состои и од *web* архитектура со цел манипулација на податоците, поточно ќе се користи *resource-oriented* архитектура која ќе користи HTTP методи (GET,PUT,POST,DELETE) во комуникацијата со серверот што значи нашиот проект исто така треба да има имплементирано и *клиент-сервер* архитектура (*network-centered style*).

Апликацијата ќе се состои и од *remote-invocation* архитектура каде што клиентот ќе повикува методи до сервисот.

Корисничката интеракција на апликацијата ќе се остварува преку *GUI архитектура (MVC pattern)* каде што моделот ќе комуницира со слојот за пристап на податоци (*податочен слој*).

Освен data access layer, оваа апликација во рамките на слоевитата архитектура ќе опфаќа и презентациски и апликациски слој.

Крајната архитектура на оваа апликација ќе биде *хибридна архитектура* која ќе претставува спој на претходно опишаните архитектури но и *дистрибуирана архитектура* со *микросервиси* кои ќе го одвојуваат корисничкиот интерфејс од серверските компоненти, и ќе овозможат поголема функционалност..

Key words

Data-flow – филтрирање на податоци
Data-centered – запишување на податоците во база
Resource-oriented – манипулација на податоци преку HTTP
Network-centered – клиент/сервер
Remote-invocation – иницирање на метод до сервисите
GUI – комуникација на моделот со податочниот слој
Layer – слоевита архитектура (презентациски,апликациски,податочен)
Heterogeneous – повеќе архитектури
Distributed arch – микросервиси-директен пристап до сервисите