Дизајн на архитектурата

Документов служи за давање приказ за архитектурата која планираме да ја имплементираме во нашата веб апликација. Тоа ќе го направиме преку неколку различни погледи за архитектурата – концептуална архитектура, имплементациска архитектура и извршна архитектура.

Апликацијата ја изработуваме во програмскиот јазик Јава и во основа ќе претставува клиент-сервер архитектура во која килентот ќе пристапува до серверот на одредена порта. Ќе се користи PostgreSQL база на податоци за да се зачувуваат податоците од користењето на апликацијата. Во неа ќе се чуваат информации за корисниците, листите од места кои ги посетиле и листите од места кои сакаат да ги посетат. Финалната архитектура на апликацијата ќе биде хибридна.

1. Концептуална архитектура

   Концептуалната архитектура на некој софтвер служи за идентификување на одговорностите на компонентите на ниво на домен, односно компонентите се поделени по концепти. Прв чекор е откривање на главните концепти од функционалните и нефункционални барања што беа поставени, а тие се:

   a. Корисник
   b. Најава
   c. Регистрација
   d. Локација
   e. Информација
   f. Посета
   g. Листа
   h. Планирано патување
   i. Додавање во листа
   j. Мапа
   k. Општина/град
   l. Пофалби/поплаки
   m. Кориснички податоци за автентикација
   n. Форма
   o. Навигациска алатка
   p. Екстерен сервис

   Следниот чекор е мапирање на сите концепти во соодветни категории (data, system, stakeholder, function, hardware, abstract concept)

   a. Корисник - stakeholder

b. Најава - function
c. Регистрација - function
d. Локација - data
e. Информација - data
f. Посета – data
g. Листа - data
h. Планирано патување – data
i. Додавање во листа - function
j. Мапа - data
k. Општина/град – data
l. Пофалби/поплаки - function
m. Кориснички податоци за автентикација - data
n. Форма - abstract concept
o. Навигациска алатка - abstract concept
p. Екстерен сервис - system

Следно, од овие поделби ќе ги извлечеме компонентите кои ќе бидат нацртани во концептуалната архитектура. Тоа се:

a. AppUI – Прикажува места за посета, како и информации за нив при кликање.
b. UserManagement - Менаџира со профили на корисници (регистрира, најавува или одјавува корисник), автентикација и авторизација.
c. Search – Пребарување на места.
d. ListsService – Менаџирање на содржини на листи со места и нивен приказ.
e. VisitedService – Додавање, бришење, менување на планирани патувања.
f. Database – Пристап до податоци, зачувување на податоци.
g. GPSService – Пристап до GPS локацијата на корисникот за автоматско додавање на местото како посетено кога корисникот е на таа локација.
h. FeedbackForm – Оставање фидбек за системот.
i. MoreInformation - Информации за местата.
j. NavigationService - Навигирање на кориснички интерфејс.

2. Извршна архитектура

Архитектурата за извршување на една веб-апликација ја опфаќа основната структура и процесите кои овозможуваат апликацијата да функционира непречено. Тоа вклучува различни компоненти и интеракции за справување со барањата на корисниците, обработка на податоци и испорака на содржина.

Во дијаграмот на извршна архитектура, најпрво ги идентификуваме компонентите на извршната архитектура, а тоа се GUI, Service, листа на посетени патувања, листа на планирани патувања, базата на податоци и надворешниот систем со кој веб апликацијата комуницира (во нашиот случај тоа е навигациски систем, на пр. google.com/maps). Во дијаграмот на концептуалната архитектура, визуелно ја претставуваме истата, идентификувајќи која група компоненти се

мапирани во архитектурата за извршување. Покрај тоа, постојат три дијаграми кои го претставуваат однесувањето на извршување кои опишуваат три случаи и тоа:

- Корисникот наоѓа повеќе информации за местата.
- Корисникот додава место во листа на планирани патувања.
- Корисникот додава место во листа на посетени патувања.

3. Имплементациска архитектура

Имплементацискиот поглед прикажува како системот е поделен на имплементациски подсистеми и елементи, односно датотеки, парчиња изворен код, интерфејси и податоци.

Апликацијата ќе има презентациски слој за кој ќе се користи Bootstrap и по потреба React, кој ќе ги прикажува податоците, односно мапата, местата на мапата, итн. Потоа, апликацискиот слој ќе биде имплементиран преку Spring развојната рамка, а во податочниот слој ќе се користи PostgreSQL како DBMS. Слоевите ќе бидат приспособени на MVC (model view controller) шаблонот.

4. Архитекурни стилови

Во оваа точка ќе ги идентификуваме главните дизајн стилови коишто ќе ги користи нашата апликација.

a. Клиент-сервер

Нашата апликација ќе имплементира клиент-сервер архитектура бидејќи во основа ќе се сведува на комуникација на клиент-корисник со централен сервер кој ќе му опслужува информации.

b. Pipe-and-filter

Нашата апликација ќе го користи концептот на цевки и филтри бидејќи тој ќе се употребува за процесирање на податоците за местата добиени во сирова форма, за да можат преку филтрите да се поправат евентуални грешки, да се добие правилно форматирање и на крајот да се претворат во објекти погодни за користење во апликацијата и за приказ во неа.

c. Notification

Нотификацискиот стил ќе биде имплементиран преку автоматската детекција на посетена локација, односно кога според GPS-от корисникот ќе се наоѓа на одредено место, системот добива “нотификација” од соодветниот сервис дека местото треба да се додаде на листата посетени места за тој корисник.

d. Microservice

Апликацијата ќе биде поделена на неколку одделни микросервиси, на пример за најава, пребарување, работа со листи и планирани

патувања итн. Тие микросервиси меѓусебно ќе комуницираат по потреба.

e. Data-centric

За секој корисник ќе се врзуваат конкретни податоци за местата кои ги посетил, местата кои сака да ги посети и неговите планирани патувања и сето тоа ќе се чува во база на податоци и ќе се обезбеди интегритетот на тие податоци.

f. Model-View-Controller

i. Апликацијата ќе биде изработена според MVC шаблонот.