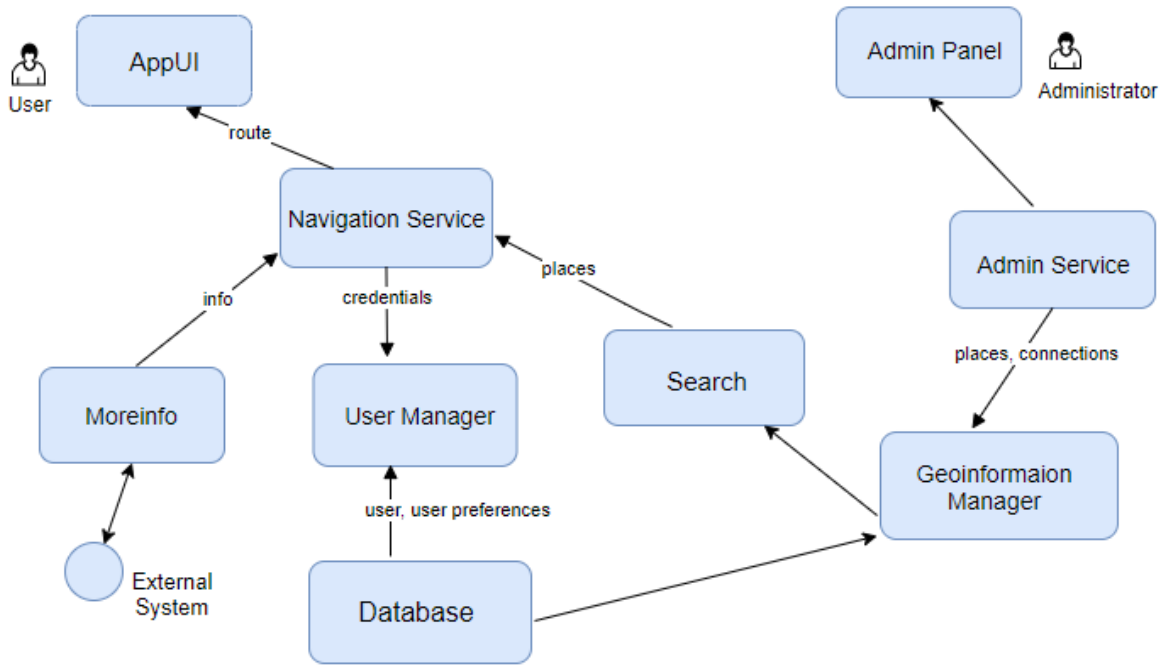
Архитектурен дизајн

Во овој документ за архитектурниот дизајн на нашата апликација ќе ви ги прикажеме концепциските, извршниот и имплементацискиот поглед кои ќе го претставуваат финалниот поглед на нашата апликација. Крајниот резултат од нашата апликација ќе претставува хибридна архитектура, која како подархитектурни компоненти ќе има цевка и филтер(за добивање на потребните податоците за станиците за полнење од open street map и нивна трансформација во потребна csv датотека) , слоевита веб архитектура(за успешна комуникација на клиентите преку нивните browsers и нашите веб и апликациски серевер) како и дистрибуирана архитектура со микросервиси и контејнеризација.

Концептуален дизајн

Од поглед на концептуалниот дизајн имаме намера да прикажеме како се распоредени одговорностите на компонентите,како компонентите се поврзани со конекторите и каков е текот на самите податоци.Концептуалниот дизајн на апликацијата можете да го погледнете на следнава слика:



На сликава може да се забележи дека имаме повеќе значајни концепти за успешно функционирање на апликацијата. Одговорностите на овие концепти ќе ги прикажеме подолу:

AppUI
-Ја прикажува апликацијата на корисникот

Admin Panel
-Ја прикажува апликацијата со додатни функционалности за администраторот

AdminService
-Ги овозможува различните функционалности кои може да ги прави администраторот

UserManager
-Овозможува:

- Регистрација на корисник
- Валидација на корисник
- Промена на лични податци
- Бришење на корисничка сметка

GeoInformational manager
-Пристапува до податоците од база
-Комуницира преку административниот сервис со администраторот
-Му овозможува пристап на search концептот до податоците за станиците

Search
- Пребарување на станици за полнење

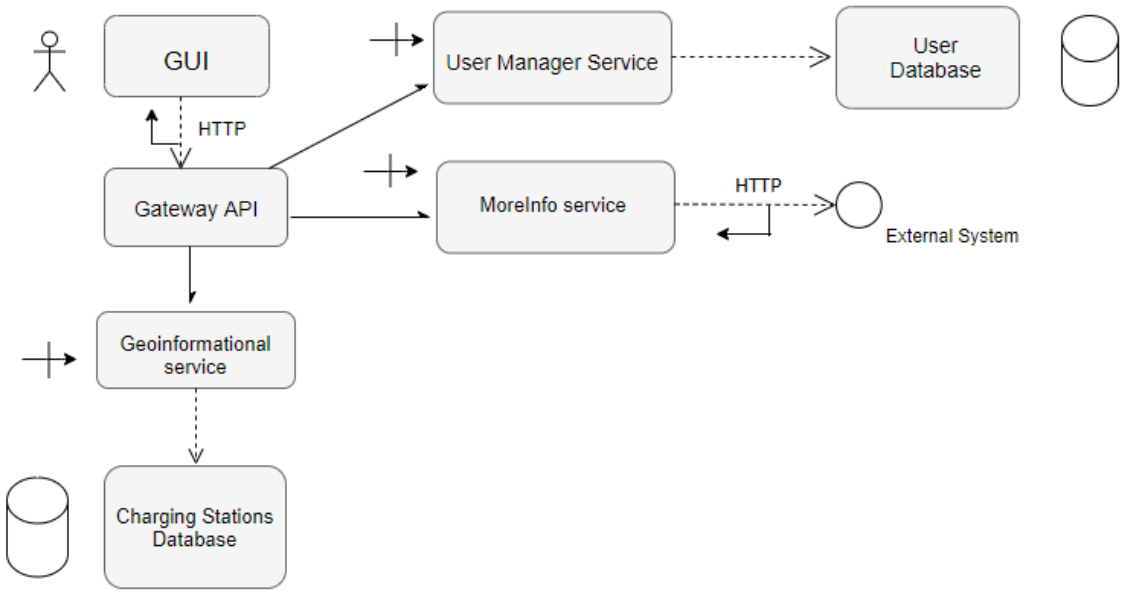
Navigation service
-Комуницира со search, user manager и more info компонентите со што ги подготвува податоците за App Ui

More info
-Овозможува дополнителни информации во врска со Станиците за полнење

Database
-Чува податоци за станиците и корисниците

Извршен дизајн

Извршниот дизајн на нашата апликација ќе ви го прикажеме со следнава слика:



Во оваа апликација планираме да имаме два конкурентни подсистеми т.е. веб прелистувач и веб сервер, при што сите имплементации ќе бидат извршени на серверска страна. Па така веб серверот е всушност серверот за веб апликацијата.

Во оваа микросервисна архитектура клентот ќе има интеракција со **3 сервиси** и поради тоа ќе користиме gateway API

Geoinformational service ќе има задача да ги извршува пресметките со податоците од станиците за полнење и има пристап до истите во соодветната база (Charging station Database).

User manager service ќе има задача да врши валидација на внесените податоци од страна на корисникот при негова најава и регистрација за што ке комуницира со соодветната база (User database).

More info service кој би имал задача да биде во интеракција со надворешните системи.

Во сиситемот имаме и **две бази** и тоа **Charging station database**(во која се чуваат податоците за станиците) и **User database**(во која се чуваат податоците за корисниците).

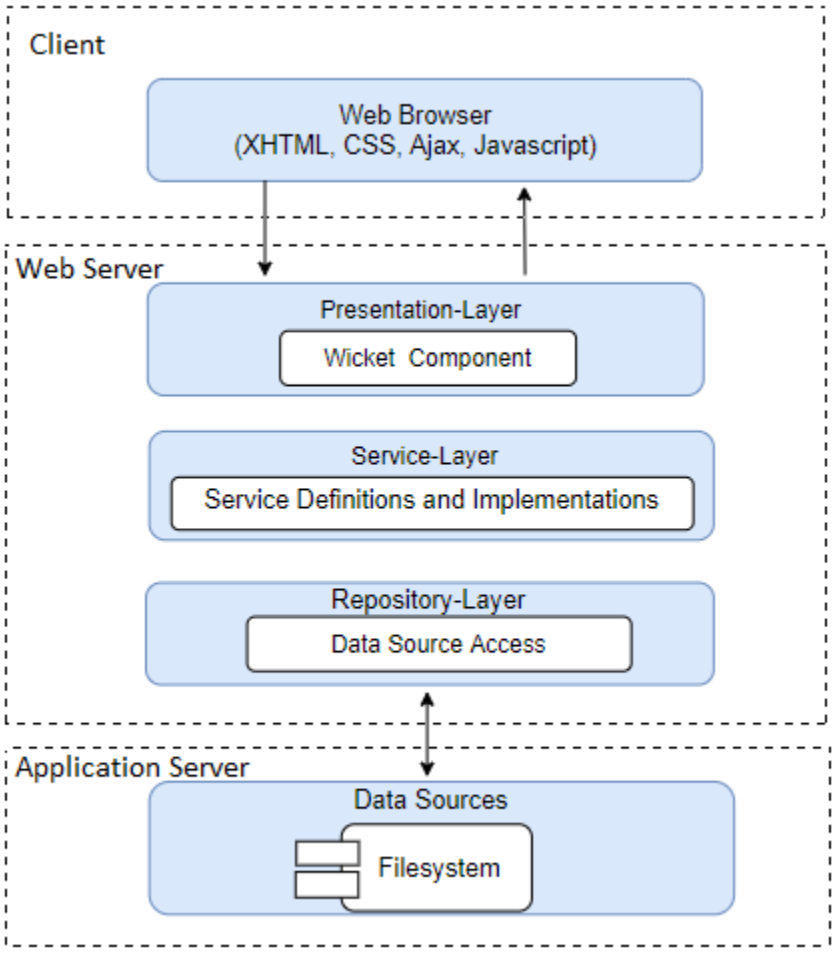
Во поглед на стереотипитите тука би имале

Иницијација од страна на корисник(User-initiated)- Кај **GUI компонентата**(Со оваа компонента ќе се прикаже графичкиот кориснички интерфејс) која ќе се користи од страна на корисникот.

Услуги(Services)-За секој од сервисите претходно ја опишавме нивната улога

Имплементациски поглед на апликацијата

На следнава слика ќе го прикажеме нашиот имплементациски дизајн:



Во овој дел ќе објасниме за тоа како системот е изграден, односно ќе кажеме повеќе за технолошките елементи кои ќе бидат имплементирани, за софтверските пакети, библиотеки..

За реализација на оваа .NET апликација од софтверски пакети планираме да го искористиме NuGet пакетот верзија 6.1.3 (последна стабилна верзија) како алатка која ќе ни овозможи поширок спектар на можности. Ќе го користиме програмскиот јазик C#.

Апликацијата ќе може да се користи неограничено.

Како апликациска компонента ќе ја искористиме HTTP Connection Handler , која претставува крајна точка со која ќе се поврзуваат повеќе конекции.

Додека пак, за инфраструктурни компоненти ќе искористиме за оваа апликација ASP .NET MVC framework, и како генерички клиент би имале пребарувач преку кој клиентите би можеле да ги најдат соодветните локации на станиците за полнење.

Серверот кој ке ни биде за контејнеризација кој ќе го искористиме ќе биде Microsoft Azure кој нуди поддршка за хибридна платформа

Кога станува збор за интерфејсот, кој исто така е дел од извршната архитектура, претставен е со комбинација на HTML/HTTP. Бидејќи ставаме фокус на интеракцијата со корисниците, ние ќе искористиме GUI Architecture, односно ќе работиме со MVC. Па така и контролерот и погледот (view) ќе бидат дел од корисничкиот интерфејс.
Конекторите, кои претствавуваат uses релација, ни се претставени со Network Protocol, бидејќи имплементациските компоненти живеат на различни процеси на различни машини. Затоа за оваа цел ние ќе користиме страндардизирани протоколи, односно ќе користиме Web services и Internet Protocol. Ќе користиме и технологија за веб услуги, односно JavaScript и XML (AJAX).