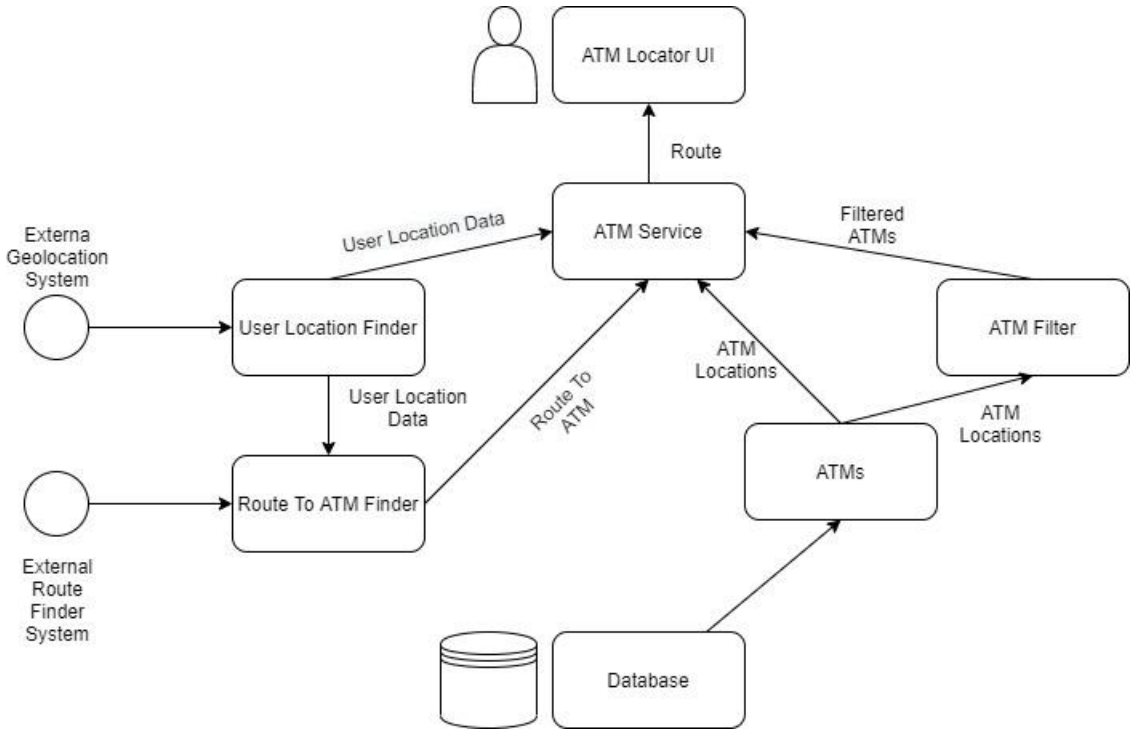
Софтверска архитектура на ATM Locator апликацијата

1. Концептуална архитектура

Со концептуалната архитектура ги прикажуваме компонентите на апликацијата и нивната меѓусебна поврзаност како и текот на податоци.

Бидејќи апликацијата ќе ја користат корисници кои треба да бидаат соодветно услужени, имаме компонента која го прикажува корисничкиот интерфејс кој ќе го гледа корисникот. Корисникот треба да може да ги гледа сите ATM прикажани на мапа, може да гледа исфилтрирани ATM според кориснички внесен филтер и рута од неговата позиција до посакуваниот ATM. Поради тоа има соодветни копоненти ATMs, ATM Filter, User Location Finder и Route to ATM Finder. Прикажување на ATM апликацијата ќе црпи податоци од соодветна база на податоци кој е прикажана како Database. Таа база ги содрши ATM машините на ниво на град скопје со одредени атрибути кои подоцна ќе ги користиме за филтрирање на истите. За координатите на корисникот решивме да користиме Geolocation API со кој на едноставен начин можеме да дојдеме до координатите на корисникот кои истите ќе се употребат за пронаѓање на рута до посакуваниот ATM. За пресметување на рутата до посакуваниот ATM решивме да користиме микросервис кој ќе ја врши таа пресметка бидејќи е покомплексна и може да наиде на промени. Сето ова е прикажано на следната архитектура.



За сето ова да го добиеме, се раководевме според SRS документот, односно функциските барања.

ATMLocatorUI:

- DisplayAllATMs
- DisplayFilteredATMs
- DrawRouteOnMap
- ShowUserLocationOnMap

ATM Service:

- CollectFillteredATMs
- CollectATMLocations
- CollectRoute
- CollectUserLocation

User Location Finder:

- GetCurrentLocationOfUser

Route To ATM Finder:

- ComputeRoute

ATM Filter:

- SearchByStreetName
- SearchByBankName
- SearchForATMsWithWheelchairOptions
- SearchForATMsWithDriveInOptions

ATMs:

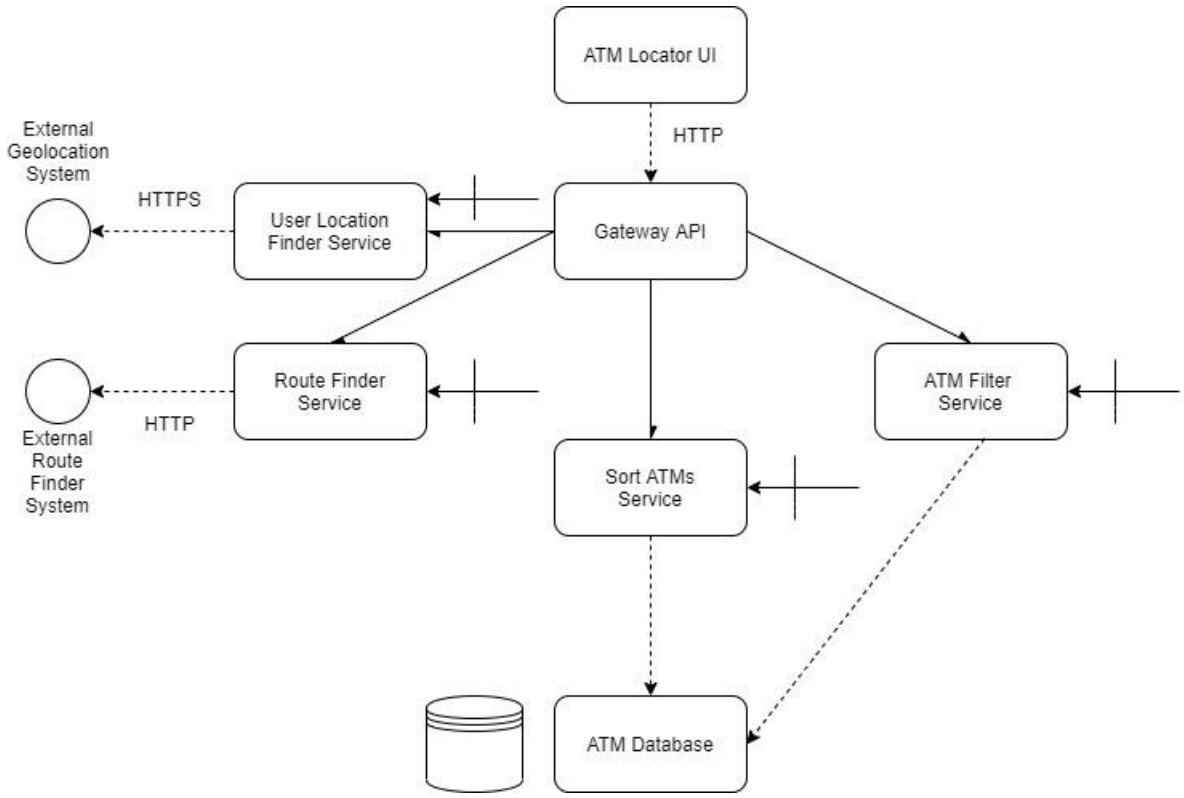
- GetAllATMs

2. Извршна архитектура

За нашата апликација Concurrent subsystems кои се користат се Web browser и Web server, со што всушност Web server-от е сервер на нашата апликација. Во поглед на стереотипите во апликацијата би имале:

- Иницијација од страна на корисник- поради пребарувањето на ATM машините поред внесениот атрибут од страна на корисникот.
- Услуги- системот чека да добие посакуван атрибут од корисникот се со цел да може да изврши филтрирање. Исто така ќе чека и дозвола за пристам до неговата локација за да може да пресмета соодветна рута до посакуваниот бакомат.

Извршната архитектура соодветно е прижана на следната слика.



3. Имплементациска архитектура

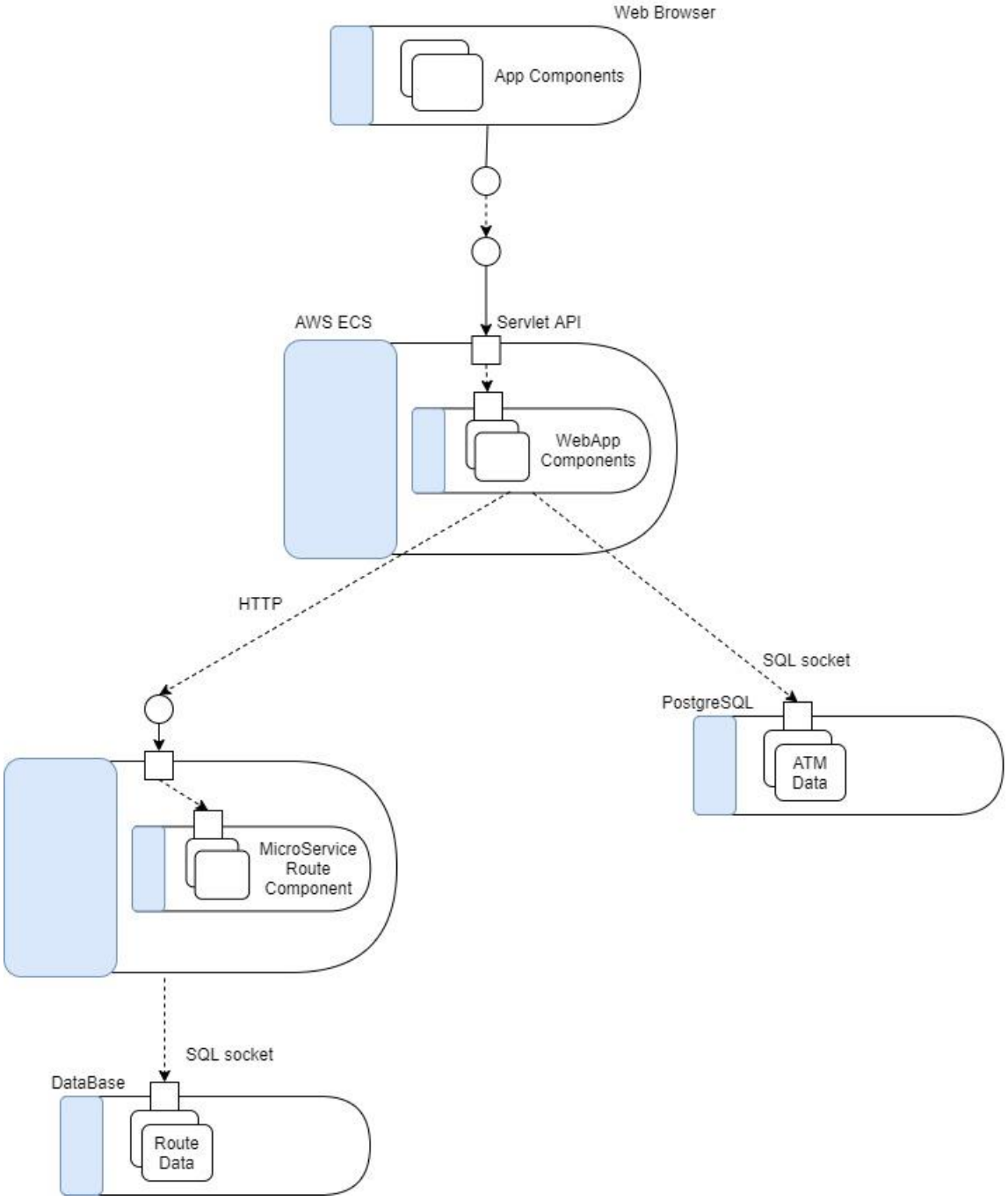
Овде треба да објасниме како нашата апликација ќе биде изградена, кои елементи ќе се имплементираат, кои библиотеки ќе се користат и слично.

Нашата апликација ќе биде .NET апликација заедно со програмскиот јазик C#. Од софтверски пакети планираме да ги користиме nuGet package manager, поточно package entity framework downloadiran со помош на nuGet package manager, Package Manager Console Host Version 5.5.0.6473.

Инфраструктурни компоненти ќе ни биде ASP .NET MVC framework-от, а апликациска компонента би ни бил пребарувачот кој корисникот ќе го користи за да ги најде соодветните ATM машини.

Бидејќи акцент ставаме на корисничкото искуство со оваа апликација и нивната интеракција, архитектурскиот стил на оваа апликација е GUI Architecture, затоа и ќе работиме со MVC. Соодветно контролерот и погледот ќе се дел од корисничкиот интерфејс. Ќе користиме дистрибуирана архитектура со микросервиси и контејнеризација. Поради тоа ќе користиме AWS ECS од Amazon која е container management service. Исто така ќе користиме и docker container за да можеме потоа да ја инсталираме нашата апликација во облак, односно на AWS ECS.

Имплементациската архитектура соодветно е прикажана на следнава слика.


Web Browser
App Components
AWS ECS
Servlet API
WebApp
Components
HTTP
SQL socket
PostgreSQL
ATM
Data
MicroService
Route
Component
SQL socket
DataBase
Route
Data