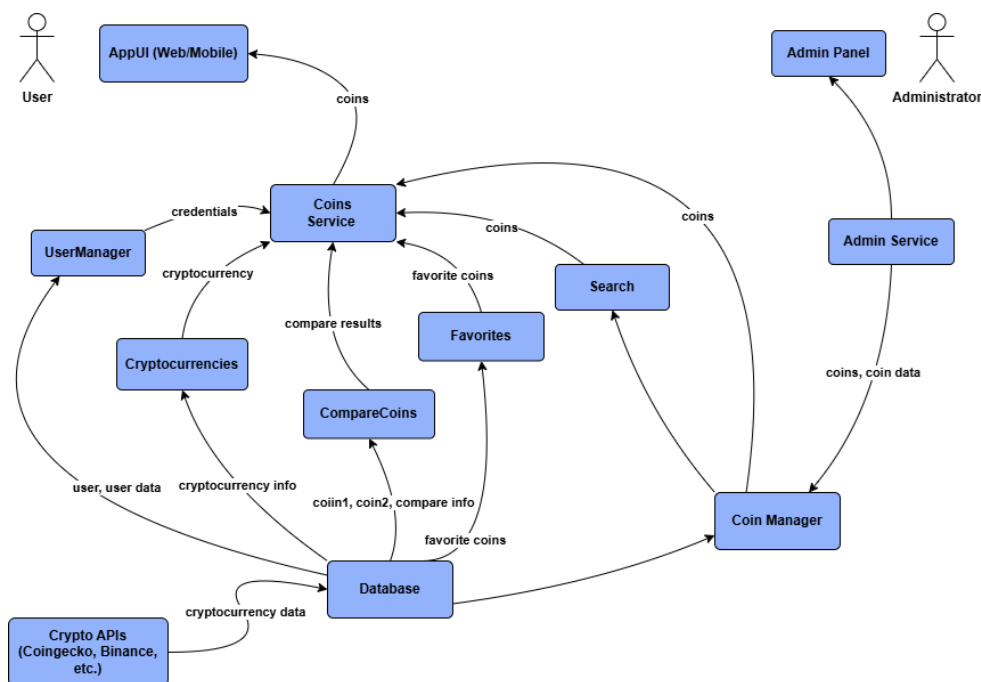


# Архитектурен дизајн

## Концептуална Итерација

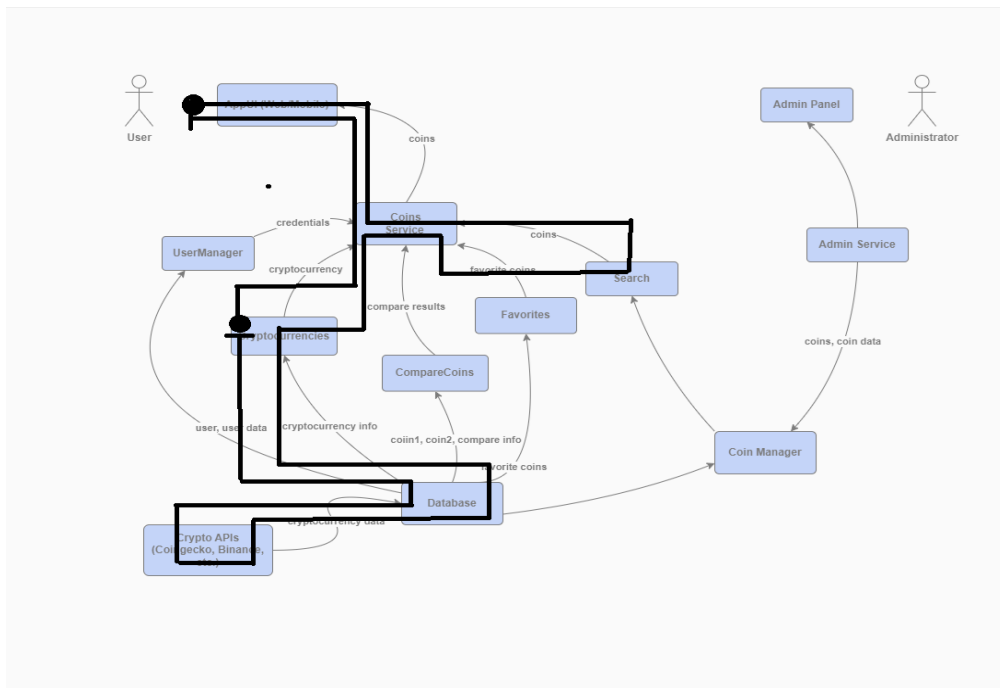
Овој архитектурен дијаграм ги прикажува сервисите и главните функционалности на апликацијата за криптовалути. Актери во системот се корисниците (потенцијални инвеститори) и администраторите. Корисниците пристапуваат до системот преку Web или мобилна апликација, додека администраторите управуваат со системот преку Admin Panel. Централен сервис во системот е Coins Service, кој управува со податоците за криптовалути и претставува посредник помеѓу корисничките функционалности и останатите сервиси. Тој комуницира со Search, Favorites, CompareCoins, Cryptocurrencies, како и со Coin Manager и UserManager. UserManager управува со корисничките податоци и креденцијали, додека Coin Manager, под контрола на Admin Service, овозможува административно управување со податоците за криптовалути. Податоците за криптовалути се преземаат од надворешни Crypto APIs (како CoinGecko, Binance и др.) и се складираат во база на податоци, од каде што потоа се користат за прикажување информации, споредби и листи со омилени криптовалути.



## Behavioral Exploration 1 (Истражување на Однесувањето 1)

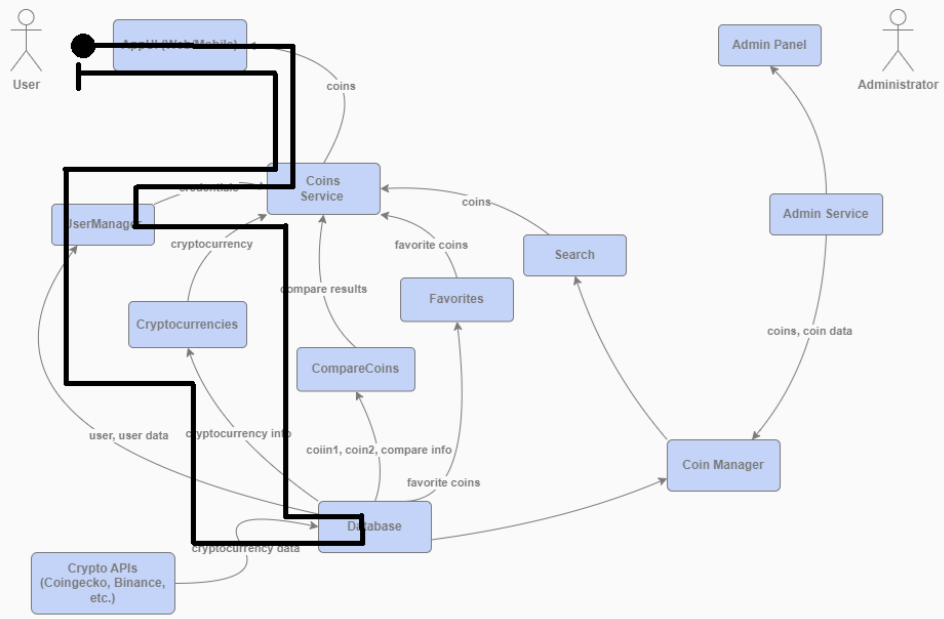
Во овој дијаграм се опишува едно однесување на системот, односно пребарување на криптовалути и соодветно прикажување на резултатите кон корисникот. Корисникот пристапува до системот преку Web или мобилна апликација и ја внесува криптовалутата за која е заинтересиран. Ова барање го активира Coins Service, кој претставува централна

компонента во системот и е одговорен за понатамошна обработка на барањето. Coins Service го проследува барањето до соодветниот сервис задолжен за пребарување, во овој случај Search сервисот. Search сервисот ја извршува својата функција со тоа што ги презема потребните податоци од базата на податоци, каде што претходно се зачувани информациите добиени од надворешните Crypto APIs. По добивањето на резултатите, Search сервисот ги враќа податоците назад до Coins Service. Потоа, Coins Service ги обработува резултатите и ги проследува кон апликацијата, по што крајниот резултат му се прикажува на корисникот во Web или мобилната апликација.



## Behavioral Exploration 2 (Истражување на Однесувањето 2)

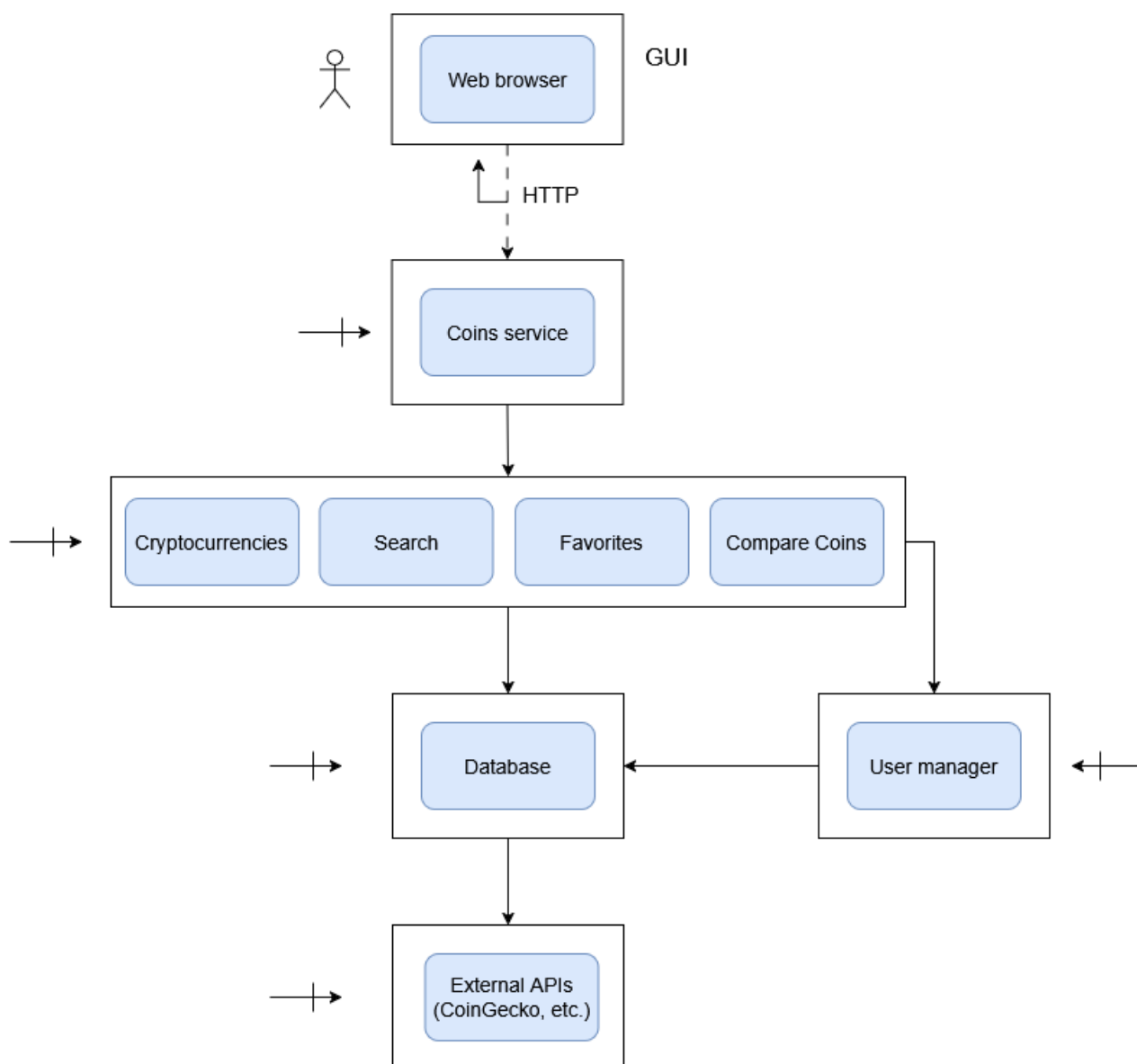
Во овој дијаграм се прикажува однесувањето на системот при регистрација и најава на корисник. Корисникот пристапува до системот преку Web или мобилна апликација и избира опција за регистрација или најава. При регистрација, корисникот ги внесува своите лични податоци (корисничко име, е-пошта, лозинка и слично). Овие податоци се испраќаат до Coins Service, кој го проследува барањето до UserManager сервисот. UserManager сервисот е задолжен за проверка на валидноста на податоците и за креирање на нов кориснички профил. Потоа, корисничките податоци се зачувуваат во базата на податоци. По успешната регистрација, се враќа повратна информација до Coins Service, а потоа и до апликацијата, со што корисникот добива потврда дека регистрацијата е успешно извршена. При најава, корисникот ги внесува своите креденцијали (корисничко име и лозинка). Податоците повторно се испраќаат до Coins Service, кој ги препраќа до UserManager сервисот за верификација. UserManager ги споредува внесените податоци со оние зачувани во базата на податоци. Доколку податоците се точни, системот ја потврдува најавата и му овозможува пристап на корисникот до функционалностите на апликацијата. Во спротивно, се враќа соодветна порака за неуспешна најава.



### Извршна архитектура – Опис (Дијаграм 2.1)

Дијаграмот 2.1 ја прикажува извршната архитектура (execution architecture) фокусирајќи се на тоа како системските компоненти комуницираат за време на извршување.

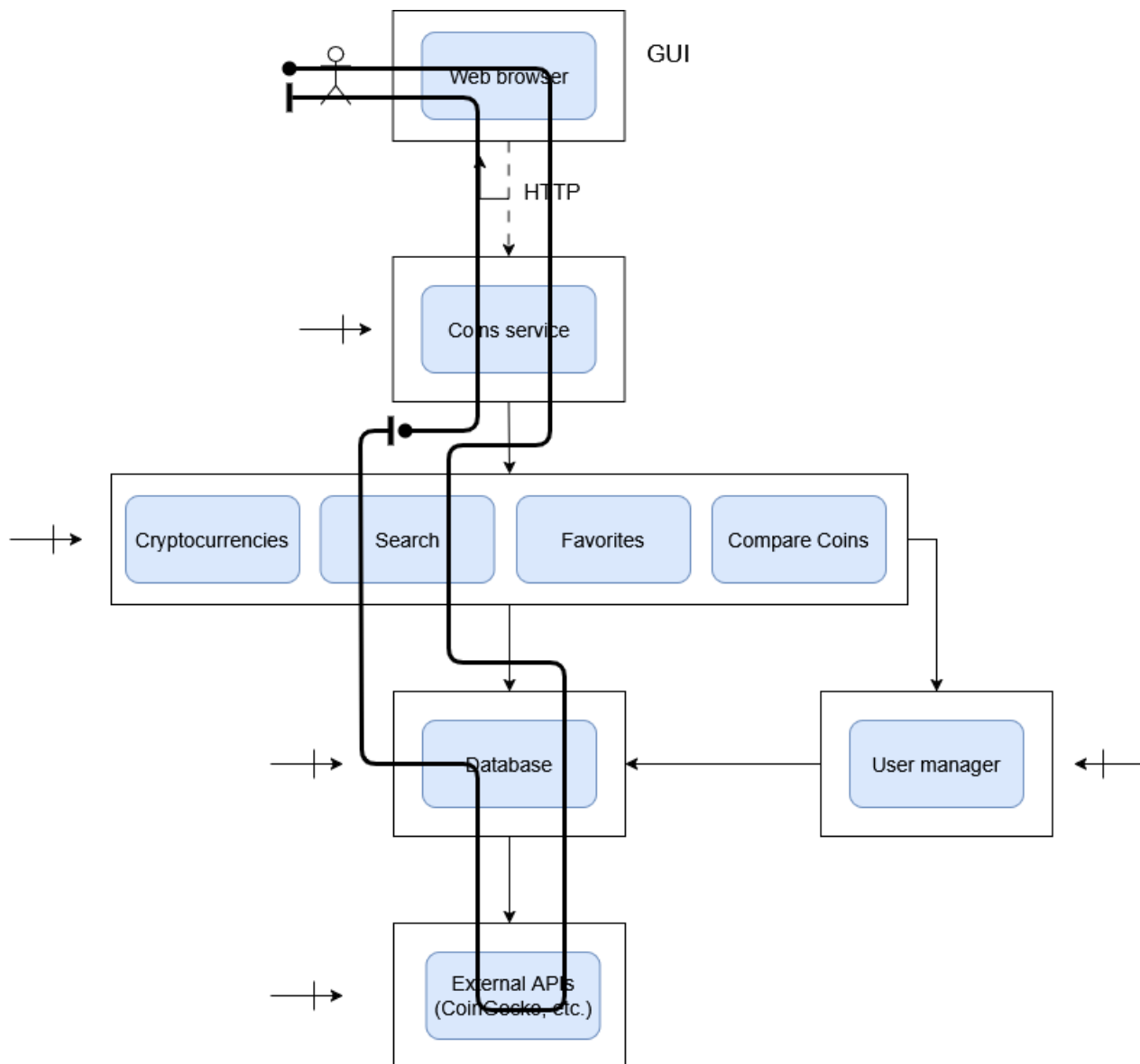
Корисникот комуницира со веб-прелистувачот, кој испраќа HTTP барања до Coins Service backend-от. Coins Service-от го координира извршувањето на функции како што се прелистување на криптовалути, пребарување, омилени и споредба на валути пристапувајќи до базата на податоци. Одделен менаџер на корисници извршува операции поврзани со корисникот, а системот презема надворешни информации за криптовалути од API-ја како CoinGecko и др. Дијаграмот го нагласува текот на извршување помеѓу компонентите за време на нормално извршување.



Дијаграм 2.1

## Извршна архитектура - Тек на извршување - Execution behaviour (Дијаграм 2.2)

Дијаграм 2.2 е behaviour use case дијаграм кој се однесува на пребарување криптовалути. Корисникот внесува име на криптовалута во веб-прелистувачот, кој испраќа барање до Coins Service. Coins Service ја проверува базата на податоци и ако податоците не се пронајдени, ги повикува надворешните API-ја за да ги преземе. Резултатот се зачувува во базата на податоци и се враќа преку Coins Service назад во прелистувачот за прикажување.



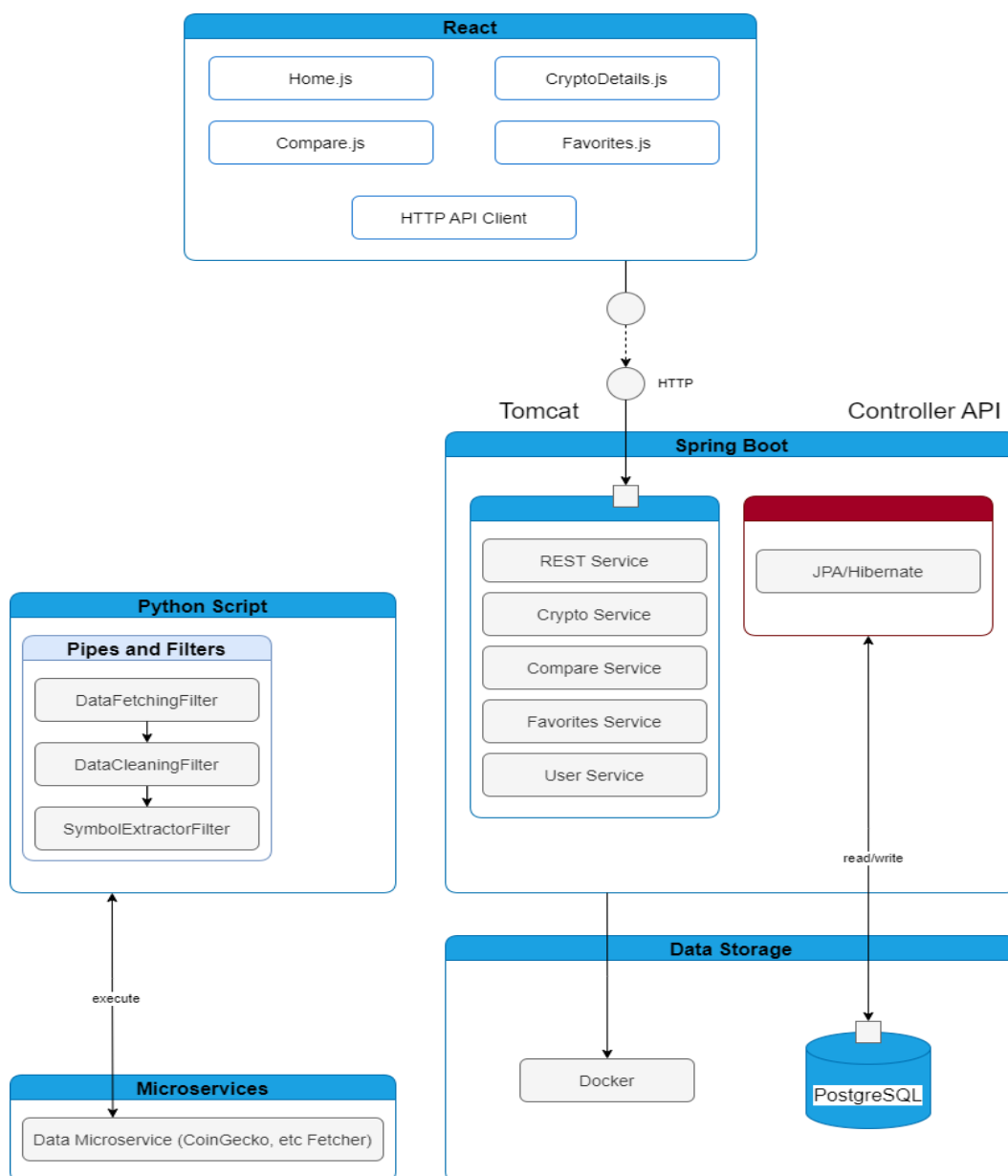
Дијаграм 2.2

### Имлементациска архитектура – Опис (Дијаграм 3.1)

Во дијаграм 3.1 е претставена имплементациската архитектура на системот, т.е. претставено е како се структурирани React клиентот, Spring Boot backend-от, компонентите за обработка на податоците и слоевите за складирање.

React апликацијата комуницира со Spring Boot REST контролерите преку HTTP врска, каде Tomcat служи како вграден апликациски сервер. Backend сервисите користат JPA/Hibernate за извршување операции за читање/запишување во база на податоци Spring Boot REST.

Дополнително е прикажан модул за Python скриптот кој ја следи архитектурата на цевки и филтри (pipes & filters) и ги обработува надворешните податоци за криптовалути добиени преку API-ја како CoinGecko и др.

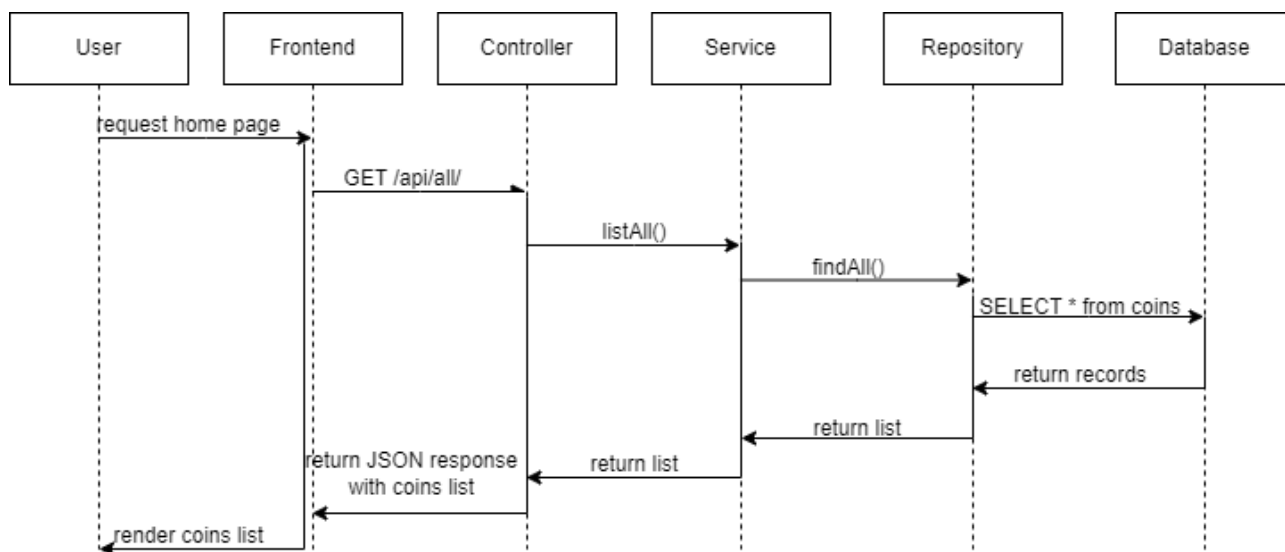


Дијаграм 3.1

### Имлементациска архитектура – Опис (Дијаграм 3.2)

Дијаграмот 3.2 претставува секвенцен дијаграм кој редоследно ја прикажува интеракцијата помеѓу ентитетите вклучени во едно барање за приказ на почетната страна на апликацијата.

Започнува со испраќање на барање од страна на корисникот (User) преку веб-прелистувачот (Frontend), кој понатаму го препраќа истото до контролерот (Controller). Тука станува збор за асинхрона порака што означува дека Frontend-от може да извршува други задачи додека чека одговор на истата. Контролерот ја делегира обработката на соодветниот Service, кој пак, комуницира со репозиториумот (Repository) за да ги преземе потребните податоци од базата на податоци (Database). Откако податоците ќе бидат преземени, тие се враќаат преку слоевите назад до корисникот во обратен редослед.



Дијаграм 3.2