Архитектурни стилови и дизајн

1. Вовед

Во овој документ се опишани архитектурните стилови и дизајнерските избори применети при развојот на системот. Главната цел на архитектурата е да обезбеди скалабилен и лесно одржлив систем.

Системот претставува N-слојна хибридна архитектура, каде што класичните слоеви (UI, сервисен слој, податочен слој) се комбинираат со микросервисна архитектура. Овој хибриден пристап овозможува јасна структурна организација и слоевитост, додека истовремено ги задржува флексибилноста и независноста што се нудат од микросервисите.

Изборот на микросервисна архитектура овозможува секоја главна функционална целина да биде независно развивана, тестирана и пуштена во употреба. Овој пристап го намалува меѓусебното влијание на компонентите и го зголемува нивото на флексибилност.

Дополнително, бекендот е контејнеризиран со Docker, со што се обезбедува конзистентна извршна средина без разлика на инфраструктурата. Ова ја олеснува локалната работа и продукциското поставување, елиминирајќи проблеми поврзани со „works on my machine“ сценарија.

Користењето на PostgreSQL како споделена база на податоци обезбедува стабилна, сигурна и добро поддржана релациона основа. Поради јасната поделба на улоги меѓу сервисите, базата се користи за конзистентно складирање на информации, додека секој микросервис има јасно дефинирани точки на пристап.

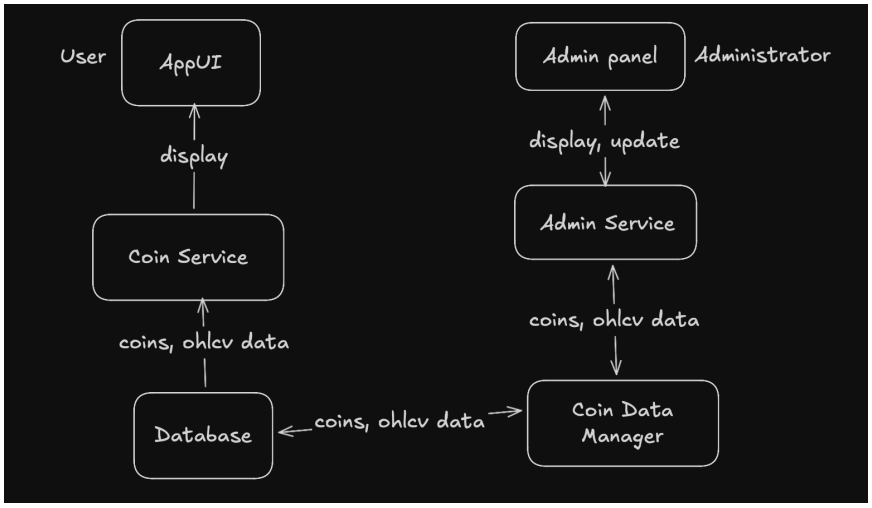
Со овој дизајн се овозможува систем кој е јасно организиран, модуларен, лесен за одржување и доволно флексибилен за идни надградби. Во продолжение следи детална анализа на концептуалната, извршната и имплементациската архитектура.

2. Концептуална Архитектура

2.1 Концептуален Дијаграм

- **Кориснички дел (лева страна):**
  - **Корисник:** Крајниот корисник на апликацијата.
  - **AppUI:** Фронтенд интерфејс (React апликација) со кој корисникот има интеракција.
  - **Coin Service:** Бекенд сервис што ги процесира корисничките барања.
  - **Database:** Заеднички слој за складирање податоци.

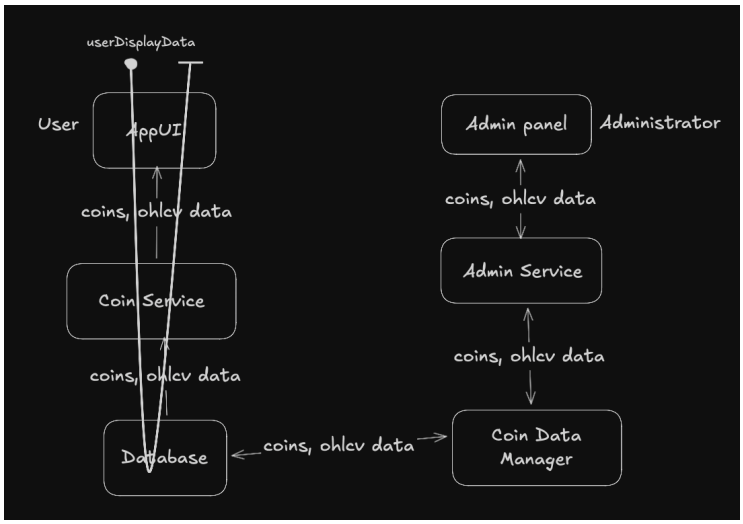
- **Администраторски дел (десна страна):**
  - **Администратор:** Системски менаџер.
  - **Admin Panel:** Специјализиран интерфејс за администрација.
  - **Admin Service:** Бекенд логиката за административни задачи.
  - **Coin Data Manager:** Специфична компонента одговорна за внесување или управување со податоци за валути во базата.
  - **Database:** Заеднички слој за складирање податоци.



2.2 Use Case Maps

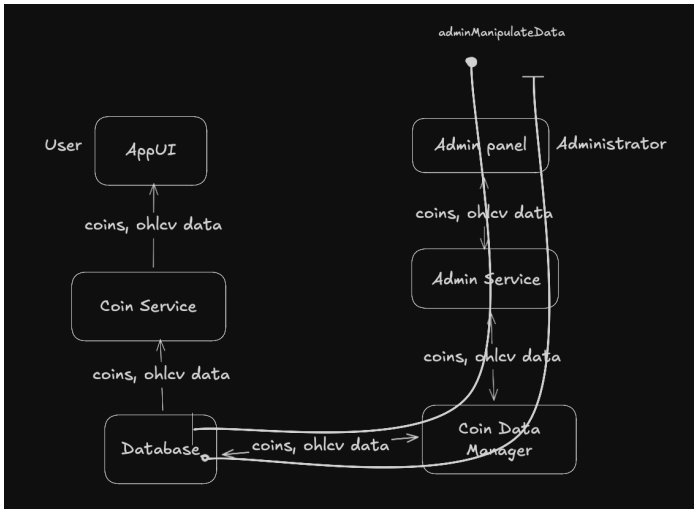
2.2.1 UserDisplayData

- **Flow:** User -> AppUI -> Coin Service -> Database.
- **Purpose:** Илустрира читање и приказ на информации од страна на корисникот.



2.2.2 AdminManipulateData

- **Flow:** Admin -> Admin panel -> Admin Service -> Coin Data Manager -> Database.
- **Purpose:** Илустрира читање, приказ и манипулација на информациите од страна на системски администратор.

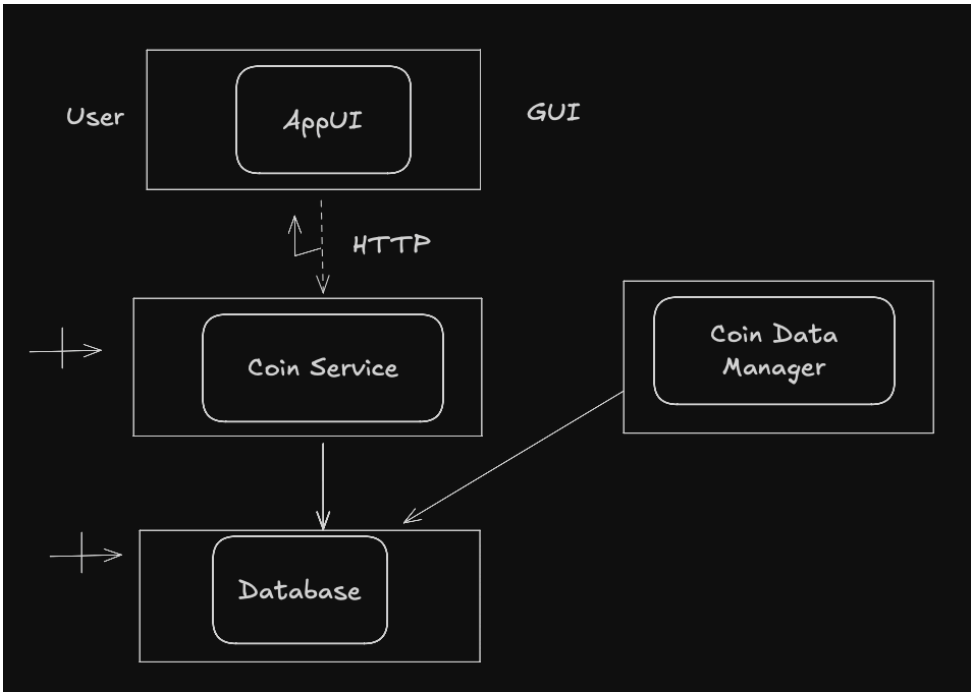


3. Извршна архитектура

3.1 Извршен дијаграм

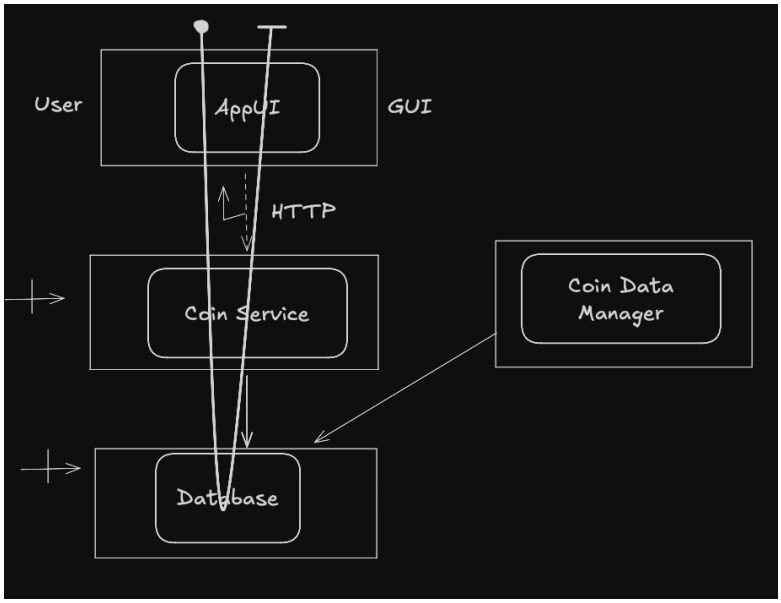
Опис на текот:

1. **User -> AppUI**
   Корисникот стапува во интеракција со графичкиот интерфејс**.**
2. **AppUI -> Coin Service (HTTP)**
   Фронтендот испраќа HTTP барање до бекендот.
3. **Coin Service -> Database**
   Бекендот зема информации од база.
4. **Coin Data Manager -> Database**
   Независно од другиот сервис, Coin Data Manager-от запишува нови податоци за валутите во истата база, со цел податоците постојано да се актуелни.
   a. Оваа компонента **не е повикана од корисникот**.
   b. Станува збор за редовен job кој се извршува еднаш дневно.



3.2 Use Case Map

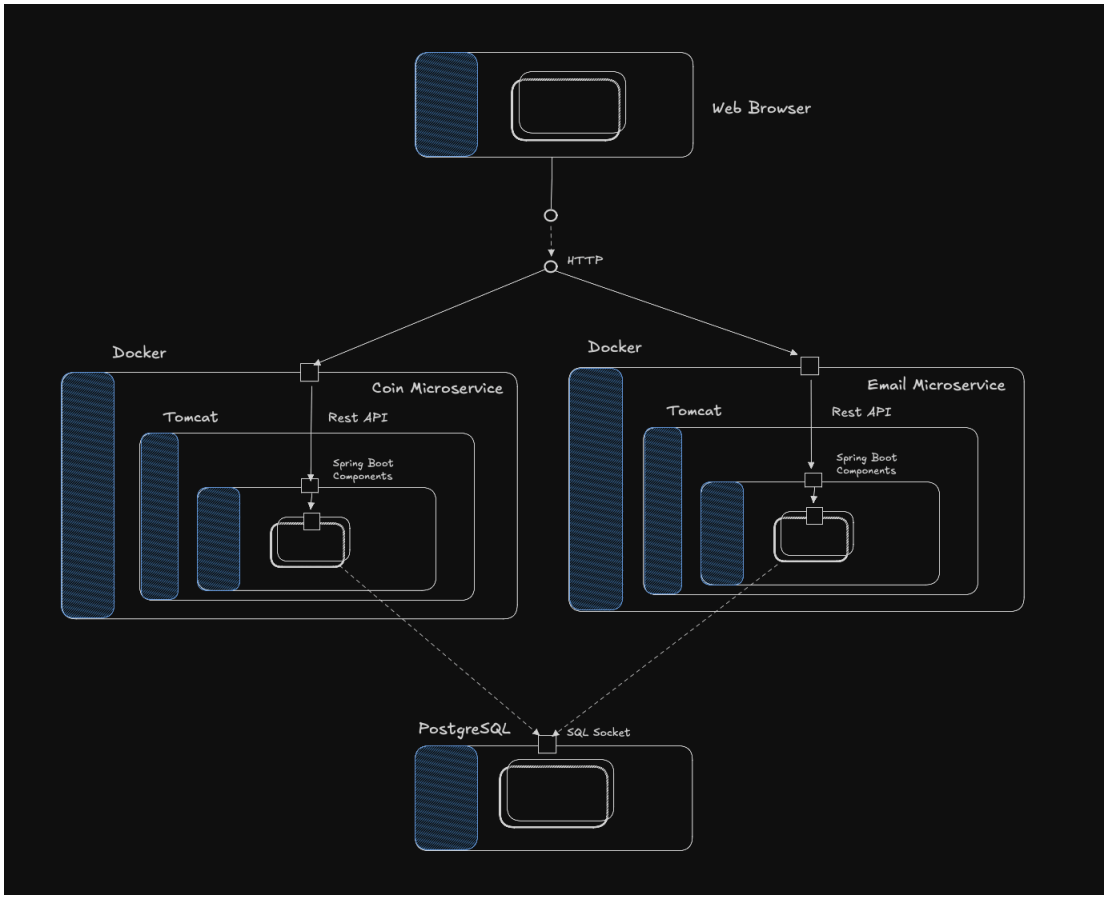
- **Flow:** User -> AppUI -> Coin Service -> Database
- **Purpose:** Илустрира собирање на податоци и нивен приказ на клиентска страна во форма на график.



4. Имплементациски архитектура

4.1 Имплементациски дијаграм

- **Docker контејнери:**
  Системот е поделен во два контејнерезирани микросервиси:
  1. **Coin Microservice:**
     - **Структура:** Користи Tomcat веб сервер, изложува Rest API и користи Spring Boot компоненти.
     - **Намена:** Основна логика за менаџирање со валути.
  2. **Email Microservice:**
     - **Структура:** Користи Tomcat веб сервер, изложува Rest API и користи Spring Boot компоненти.
     - **Намена:** Одделно од логиката на главната апликација, се занимава со испраќање на мејлови.
- **База на податоци:**
  - **PostgreSQL:** Двата микросервиси се поврзуваат до споделена Postgres база на податоци.



4.2 Секвенцен дијаграм

- **Flow**: User -> Coin Service -> DB
- **Purpose**: Опслужување на корисничко барање