**Дизаjн и архитектура на софтвер**

**Домашна работа 1**

**3. Барања**

Проектот има два основни типа на барања: **функциски** и **нефункциски**.

#### **1. Функциски барања:**

Функциските барања ги опишуваат конкретните функционалности што апликацијата треба да ги обезбеди за да ги исполни потребите на корисниците. Тие се директно поврзани со бизнис логиката на апликацијата.

1.1. Системот треба автоматски да презема листа на сите издавачи од веб-страницата на Македонската берза, освен обврзниците и сите кодови кои содржат броеви.

1.2. Системот треба да ги складира кодовите на издавачите од листата за подоцнежно процесирање.

1.3. Системот треба да проверува дали има веќе зачувани податоци за секој издавач и до кој датум се ажурирани тие податоци.

1.3.1. Системот, доколку нема постоечки податоци, треба да преземе историски податоци за последните 10 години.

1.3.2. Ако податоците веќе постојат, системот треба да го идентификува последниот датум на зачуваните податоци и да ја пренесе информацијата за понатамошна обработка.

1.4. Системот треба да ги преземе сите податоци што недостасуваат до тековниот датум за секој издавач и да ги зачува во базата на податоци или структурирана датотека (на пр. CSV, JSON).

1.5. Системот треба автоматски да форматира датуми и цени во конзистентен формат (на пр. YYYY-MM-DD за датуми и 21,600.00 за финансиски вредности).

1.6. Системот треба да зачува сите податоци на унифициран начин, осигурувајќи конзистентност помеѓу новите и постоечките записи.

1.7. Системот треба да овозможи извештај за успешноста на процесот на преземање и обработка на податоците, што ќе вклучува статистика за времето на обработка и бројот на обработени издавачи.

1.8. Системот треба да овозможи враќање на историски податоци за конкретен издавач на основа на избран период (на пример, последни 5 години или последни 30 дена).

1.9. Системот треба да има можност за конфигурирање на автоматска обновување на податоците по определен период (на пример, на секои 24 часа).

1.10. Системот треба да овозможи корисникот да избере специфични издавачи и да ги преземе само податоците за тие издавачи.

1.11. Системот треба да обезбеди API интерфејс за надворешни апликации или корисници за да можат да ги пристапат податоците за издавачите.

#### **2. Нефункциски барања:**

Нефункциските барања се поврзани со квалитетот на системот, неговата перформанса, безбедност, корисничко искуство и слично.

2.1. Системот треба да биде оптимизиран за ефикасност и да го минимизира времето на обработка на податоците.

2.2. Системот треба да биде дизајниран со pipe-and-filter архитектура за лесно одржување и интеграција на нови податоци и извори.

2.3. Системот треба да овозможи изведба на процесот на обработка во разумен рок, односно во времетраење што не го надминува оптималното време (мерено со тајмер).

2.4. Системот треба да обезбеди доследно форматирани и стандардизирани податоци за лесно читање и обработка од страна на други системи или апликации.

2.5. Системот треба да биде изработен во програмски јазик кој е соодветен за трансформација на податоци, со користење на библиотеки кои овозможуваат манипулација на финансиски податоци и датумски формати.

2.6. Системот треба да биде безбедно имплементиран, така што сите податоци на корисниците и компаниите ќе бидат заштитени од неовластен пристап.

2.7. Системот треба да биде скалабилен за да може да обработува големи количини на податоци (на пример, ако бројот на издавачи значително се зголеми во иднина).

2.8. Системот треба да биде прилагодлив за употреба на различни платформи (Windows, Linux, macOS) и да има лесно достапен интерфејс.

2.9. Системот треба да обезбеди резервни копии на податоците кои редовно ќе се прават и чуваат на безбедно место, за да се обезбеди континуитет на услугата.

2.10. Системот треба да биде лесен за одржување, со добро документиран код и јасно дефинирани процеси за додавање на нови филтри или функции.

### **Кориснички сценарија и персони**

#### **Персона 1: Финансиски аналитичар**

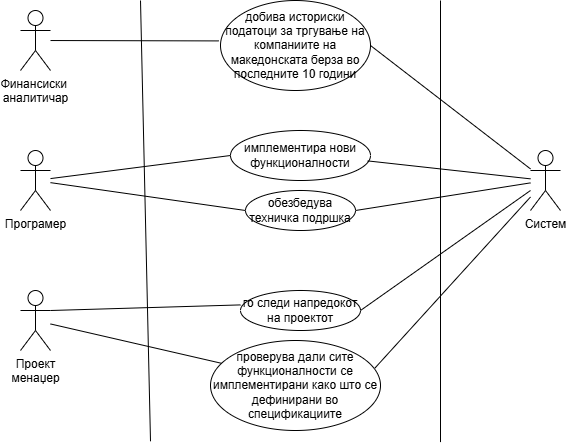
* **Сценарио:** Финансискиот аналитичар користи апликацијата за автоматски да презема и анализира податоци за различни компании. Тој сака да ја оптимизира својата работа со тоа што ќе користи податоци без да мора рачно да ги собира. На пример, аналитичарот може да го следи трендот на цените на акциите на некоја компанија и да изготви прогноза за идните трендови.

#### **Персона 2: Програмер**

* **Сценарио:** Програмерот користи Python за да дефинира филтри за преземање и обработка на податоците, обезбедувајќи дека кодот е модуларен и лесен за одржување. Тој додава нови филтри за нови извори на податоци, или за оптимизација на процесите на претходни филтри.

#### **Персона 3: Проект менаџер**

* **Сценарио:** Проект менаџерот ги следи резултатите на тимот преку GitHub, прегледувајќи документи, код и обезбедувајќи дека сите функционалности се имплементирани според зададените барања. Проект менаџерот исто така ќе ја следи ефикасноста на апликацијата преку времето за обработка на податоци.



Со оваа спецификација на барањата се дефинираат функционалностите кои апликацијата треба да ги исполнува, како и параметрите кои влијаат на квалитетот и перформансите на системот.