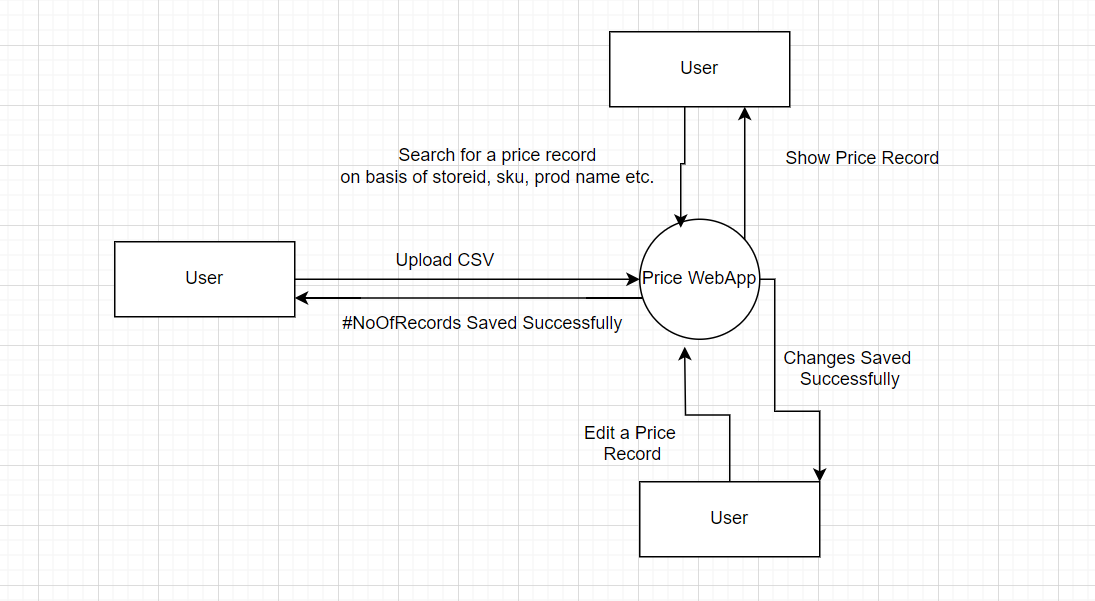
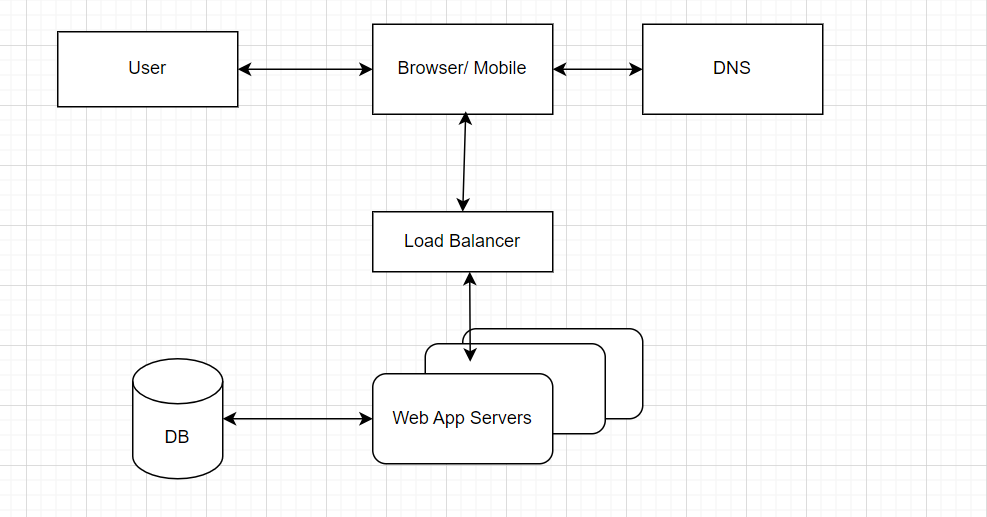
**Context Diagram:**

****

**Solution Architecture:**

****

**Design Decisions:**

**Backend is built using fastapi because of its asynchronous capabilities and high performance.**

**Frontend is built using React because of its popularity, dynamic and responsive user interface.**

**Database though currently I have used sqlite as its in-memory but we can use MySQL or SQL Server relational database to support ACID properties which ensures data consistency and also indexing can be implemented on frequently queried fields since we have to implement a functionality to search a price record.**

**To scale backend or frontend we can dockerize react app or fastapi and deploy it on multiple servers.**

**Non-functional requirements considered and how the design addresses them**

**Non-functional requirement stats that, there could be 3000 stores across multiple countries. So in order to support it the system should be able to handle huge user request and huge data volume.**

1. **We can dockerize react app or fastapi and deploy it on multiple servers.**
2. **Relational Database which supports indexing and ACID properties.**

**Assumptions**

**DataSize: 3000 (stores) \* 5000 (total SKUs) \* 1 KB (Size required for a SKU) = 15 GB**

**API Requests: 3000 (stores) \* 1000 (request per day) = 3000000 Request/ Day**

**Peak API Request: 3000000 / (12 \* 60 \* 60 ) \* 2 = 140 Request/Sec**

**Source for the implementation**

[**https://legacy.reactjs.org/docs/getting-started.html**](https://legacy.reactjs.org/docs/getting-started.html)

[**https://fastapi.tiangolo.com/**](https://fastapi.tiangolo.com/)