Iteration 3: Addressing Quality Attribute Scenario Driver

This iteration focuses on further developing the design decisions made in the previous two iterations. However, this iteration varies as it also introduces a focus on the fulfillment of some of the key quality attributes. For this iteration, the focus was placed on Quality Attribute 4: Performance.

2. Establish Iteration Goal by Selecting Drivers

The driver for this iteration was:
QA-4 Performance: a user accesses the main homepage of the website. The page and listings to the site are sent to the user and displayed (on test workbenches) within 30 seconds.

3. Choose One or More Elements of the System to Refine

The elements to be refined are the components involved with the generation of the main homepage of the website from iteration 2.

In this case, these components are the **Generate Page component** and the **Database Server component**. It should be noted that the **Request Data Component** is a secondary element that is refined during the iteration. This component was added to the list of elements refined during step 4, as the introduction of the local data cache also included the storage of data in a new file format. This means the way that pages request data will need to change to accommodate the local cache and new data format.

Step 4: Choose One or More Design Concepts That Satisfy the Selected Drivers

The design concepts chosen to help satisfy Quality Attribute 4 are summarized in the table below. The techniques chosen stem from the Performance Tactics provided in Appendix A of the textbook [1]. For the decisions made, emphasis was placed on managing resources. It should also be noted that controlling the demand for said resources also plays a large role in improving performance, and as a result of this one tactic from controlling demand was integrated into the design during this step.

Design Decisions and Location	Rationale and Assumptions
Manage Resources: Increase resources by investing in reliable database and web application host to reduce latency.	By increasing the reliability, dependability, and overall efficiency of the server setup, the speed at which each user is able to access the homepage and databases will increase.
Manage Resources: Maintain multiple copies of data by using local data cache in addition to main database.	Introducing a local cache will allow the homepage listings and user's profile and respective listings to be stored on the user's system for quick repetitive access.

Control Resource Demand: Increase resource efficiency by further researching local cache data storage formats. Improve (introduce) algorithms and techniques used to fetch most recent and relevant listings to the user.	By further researching to improve the methods in which the data is stored on the local cache, the speed at which they are retrieved can be improved. Introducing algorithms to fetch relevant listings further allows selective data to be stored on local cache, and limits calls to database. Also assists with QA-6 (Usability) by providing context aware listings on the main page and in search, improving the likelihood in which a user can find an item they wish to offer on (Improving and upholding the task model).

Step 5: Instantiate Architectural Elements, Allocate Responsibilities, and Define Interfaces

The design decisions from the previous step are allocated to direct components of the system, and specific technologies are chosen for implementing these decisions. Further rationale for the decisions made and the specifics on the implementation are also provided in the table below.

Design Decisions and Location	Rationale
Deploy frontend application using AWS Lightsail for dynamic/web hosting	Hosting the frontend application using AWS Lightsail helps minimize site downtime and simplifies the process of hosting the web application.
Deploy web application database on Amazon RDS	Hosting the database using Amazon RDS helps minimize any latency issues that stem from hosting the database server on a specific workstation. This also improves/minimizes downtime.
Local data cache on the user's system to maintain multiple copies of listing and users profile	Listings that are accessed by the user can be saved on the local data cache during each session to eliminate the need to re-fetch the information from database
Store user profile, user's listings, and listings viewed in a session on local data cache in efficient format.	Using a new data format to improve speed of storing data on local cache will also improve the rate at which this data can be retrieved. JSON format is being considered, but further research is required.
Listings stored in local cache used in existing algorithms to obtain other listings the user may be interested in upon search. If nothing in cache, use generic search.	Using a pre-existing search-optimization algorithm allows for this feature to be implemented without developing an ad hoc solution from the ground up. Apply to most recent listings on the homepage too. Again supports QA-6 (Usability) by providing context aware listings.

Step 6: Sketch Views and Record Design Decisions

This step refines the diagrams developed in previous stages/iterations of the ADD process to incorporate the new decisions made in the past two steps.

Figure 7 shows a refined deployment diagram of the system that focuses on the introduction of the new deployment technology used to create the system, as well as the locations of the tech used.

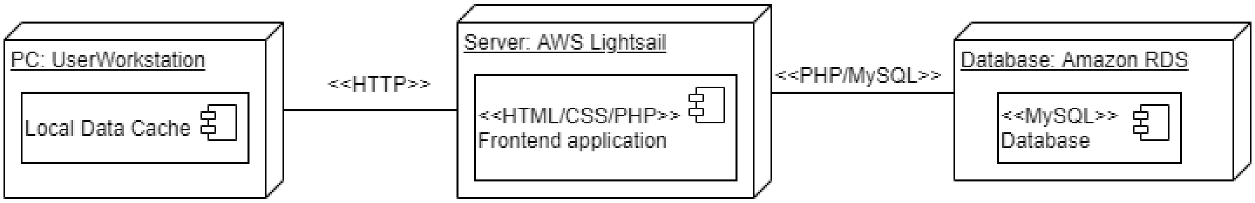

Figure 7 : Iteration 3 Revised Deployment Diagram

Figure 8 shows a UML sequence diagram of the Quality Attribute identified for this iteration, QA-4. This sequence diagram demonstrates the components used to support UC-6 and UC-13, and how they interact upon logging in to a fresh session of the system in contrast returning to the home page after browsing the site.

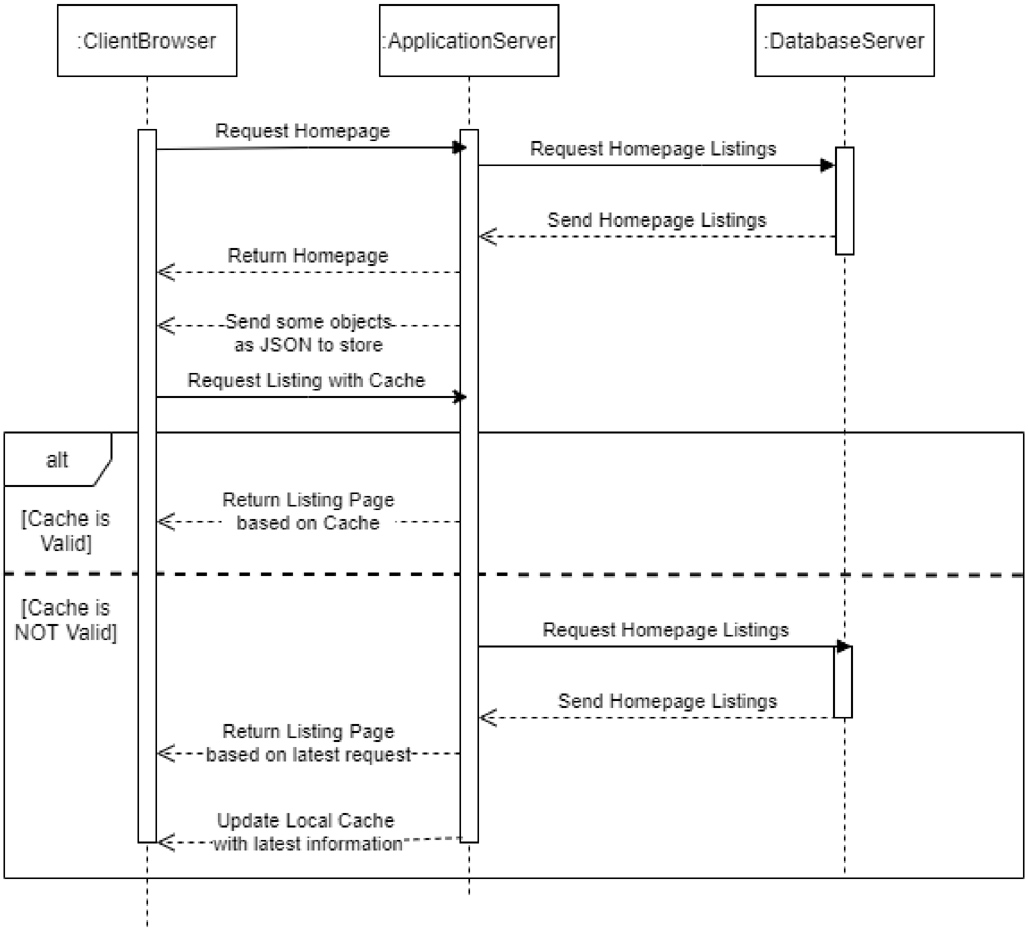
:ClientBrowser
:ApplicationServer
:DatabaseServer
Request Homepage
Request Homepage Listings
Send Homepage Listings
Return Homepage
Send some objects as JSON to store
Request Listing with Cache
alt
[Cache is Valid]
Return Listing Page based on Cache
[Cache is NOT Valid]
Request Homepage Listings
Send Homepage Listings
Return Listing Page based on latest request
Update Local Cache with latest information

Figure 8 : Sequence Diagram when accessing homepage with introduction of local cache

Step 7: Perform Analysis of Current Design and Review Iteration

The focus of this iteration was QA-4 (Performance), and numerous design decisions were made to support the driver goal. These decisions also resulted in several improvements to use cases that had been completely addressed previously. In doing so, QA-6 (Usability) has also been positively impacted as the decisions made uphold and improve the task model by providing context aware information and options to the user-- a key Usability tactic outlined in Appendix A [1].

Not Addressed	Partially Addressed	Completely Addressed	Design Decisions Made During Iteration
CON-3	CON-1, QA-3	QA-4, UC-2, CON-4	Database deployed on Amazon RDS. Frontend deployed on AWS Lightsail.
		QA-4, UC-6	Local data cache on user system
	UC-6, UC-14		Local data cache stores in efficient format, further research into JSON storage required (Specific technologies not chosen, partially addressed).
		UC-2, UC-6, UC-13, UC-14	Algorithm for search and listings and homepage display.

Quality Attributes, Constraints not addressed/applicable in this iteration:
QA-3, QA-7

References

[1] H. Cervantes and R. Kazman, *Designing Software Architectures: A Practical Approach.* Addison Wesley, 2016. [E-book] pp. 211-246