Iteration 2: Identifying Structures to Support Primary Functionality

In this iteration, decisions that determine the methods used for implementation and the development teams are made here. Concrete decisions regarding location of components, implementation technologies, and some reasoning regarding the interfaces which are required by the various components.

2. Establish Iteration Goal by Selecting Drivers

For this iteration, a few primary use cases were selected as the driving forces for system decisions:

UC-4, Listing Creation
UC-6, View Listing
UC-7, Offer Creation
UC-8, View Received Offers

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

The elements of the system chosen to be refined for this iteration are the various components needed to execute each of the use cases described in step 2, however, emphasis is placed on the components needed to achieve the basic functionality of the site: viewing and posting items.

- Browser/UI
- **Generate Page**
- **Request Page**
- **Request Data**
- Modify Data
- Data Access
- Validate login/current user

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

Some design concepts were selected to help fulfill the primary use cases and begin identifying the various domain objects and their relationships to help further guide implementation and modelling.

Design Decisions and Location	Rationale and Assumptions
Identify domain objects that relate to corresponding use cases	Domain objects are Listings, Users, and Offers. Pages interact with these objects to implement functionality. Functionality referring to viewing, creating, or modifying these objects
Create Domain Model (website map of sorts) to show where domain objects interact within the system	A domain model must be created to indicate where objects interact with the various navigation pages, which act as generators based on the object information passed to the ViewPage component. Must create a domain model, as without it functionality and architecture will not be clearly defined, resulting in a design that cannot easily be followed and maintained.
Construct modules/components from decomposed domain objects (identify key functionality and groupings/units)	Objects and pages must be grouped based on where they lie within the layered architecture. This is to create specialized components based on required functionality.

5. Instantiate Architectural Elements, Allocate Responsibilities, Define Interfaces

Functionality is allocated to the various components, and decisions regarding the interaction processes and responsibilities of each element is outlined in the table below. Some preliminary decisions regarding implementation technology are made during this step as well

Design Decisions and Location	Rationale
Create basic domain model with focus on interactions and components	Objects interact with various pages of the system to generate different content based on the navigation page selected by the user. Only a simple domain model is to created to expedite this part of the development process, as there are not many objects given the nature of the system

Objects will be stored as tuples in tables in a MySQL database	Some pages will need several specific objects to create them so they need to queryable. Objects includes: Listings (ID, Title, Description, PostingUser) Users (ID, Username, password) Offers (ID, PostingUserID, RecivingListing ID, Description, ContactDetails) ViewPage(input listings, user profile, or Offers to generate a page to display them) -> function associated with several of the objects, varies depending on page attempting to be generated
Several relevant object instances will be stored in the Local Data Cache as a JSON string. Only necessary and displayable information is stored.	Some pages require object information where the objects they may need are a smaller subset rather than the whole dataset. These objects can be predicted, such as all listings belonging to a specific user needed for easy loading from that user's profile, or the most recent/popular listings to be displayed on the homepage for quick traversal back and forth from those listings and the home page. These predicted objects will be queried from the database and stored as a Json string on the user client. When a page request is sent, the string timestamp is sent alongside to check whether new information should be queried and sent or if the existing information in the local cache store is valid for page generation. Preventing the database to be queried several times for similar queries and over reliance on the server.
Pages will be generated using PHP	PHP offers built-in methods to access a MySQL database and support for extracting information from those tuples into associative arrays alongside Json string decoding into the same array format. So the same code to generate pages can be used for both the local data and database server data.

6. Sketch Views and Record Design Decisions

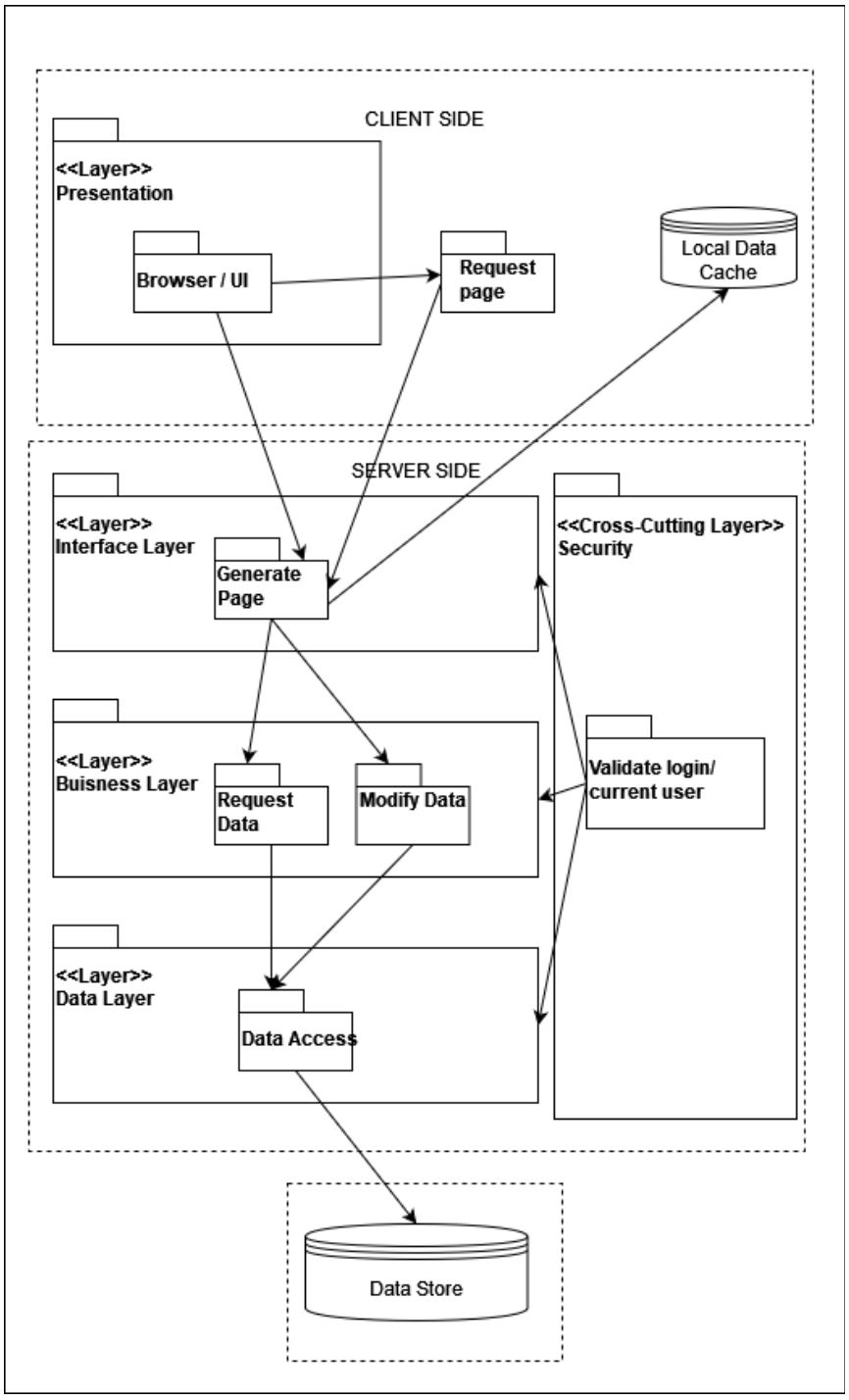

Figure 4 : Revised system diagram from iteration 1. This revised diagram shows the implementation of a local data cache and the relocation of the Request Page component. Upon page generation the local cache is updated with listings and other information required to display listings and the homepage. When a page is requested through the Request Page component, the Generate page component chooses to generate the page either by using the local data cache it was sent or by interacting with a lower layer to access the database server.

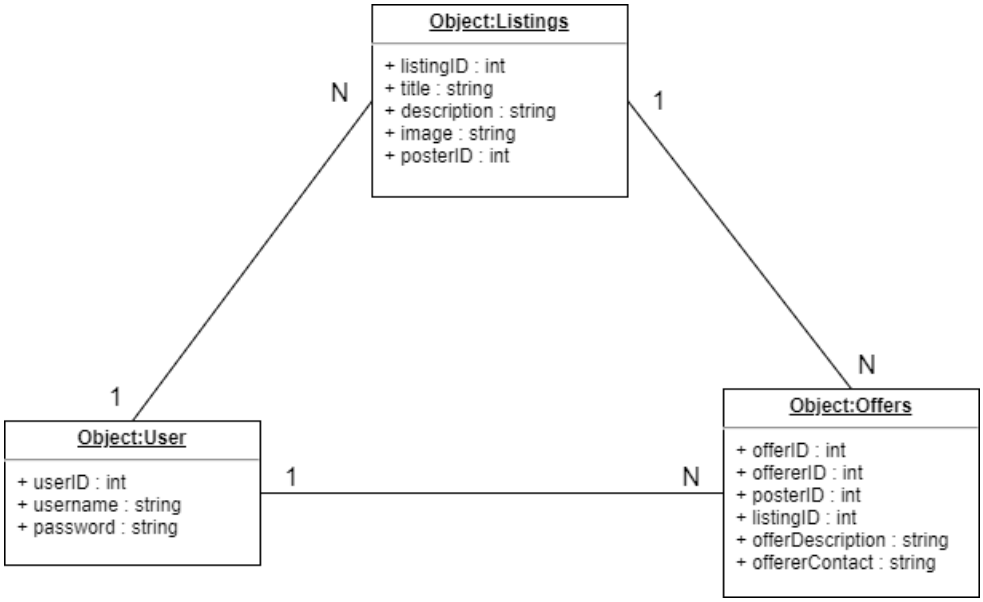

Figure 5 : Displays three kinds of entities that will be used in the system.

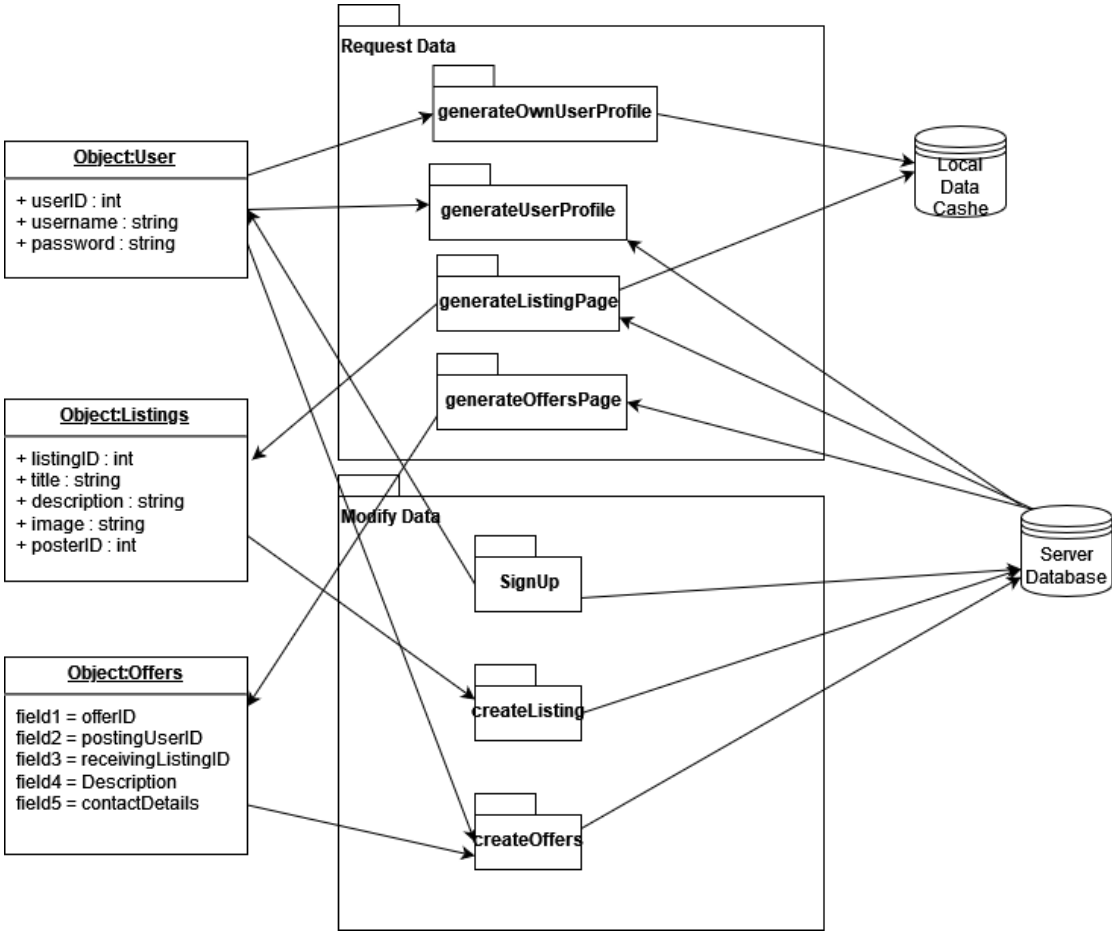

Figure 6 : This diagram shows the relations of the different domain objects to use cases and where those use cases would retrieve the needed objects from.

7. Perform Analysis of Current Design and Review Iteration Goal and Achievement of Design Purpose

The focus of this iteration were several of the primary use cases. Numerous design decisions regarding implementation technology and component locations were made to support them.

The use cases or quality attributes which no longer require further refinement are stored within the “Completely Addressed” column. Anything “Partially Addressed” should be revisited in further iterations to complete the system design to the required level of detail.

Not Addressed	Partially Addressed	Completely Addressed	Design Decisions Made During Iteration
		QA-6	Create basic domain model with focus on interactions and components
		UC-7, UC-4, UC-15	Objects will be stored as tuples in tables in a MySQL database
	QA-1	QA-4, UC-6, UC-14	Relevant objects stored as a Json string in a local data cache.
	QA-4	UC-6, UC-8	Pages will be generated using PHP

Quality Attributes, Constraints not addressed/applicable in this iteration:
QA-3, QA-7, CON-1, CON-3

The focus of this iteration were the use cases and implementing a system to clarify components behaviour in them. Design decisions were made to support this goal. Notably UC 14 was influenced in the introduction of the local data cache to store context relevant listings so that the main database server is not constantly queried with redundant searches.