**Low-Level Architecture and Data Models**

**P02:BaechDay.com**

**<team member names & ids>**

|  |  |
| --- | --- |
| **Student ID** | **Name** |
| **23100181** | **Nashit Iftikhar** |
| **23100225** | **Muhammad Arsalan Ullah Tarar** |
| **23100157** | **Moiz Nafey** |
| **23100089** | **Mahad Mubashir Beg** |
| **23100334** | **Silal Anwar** |

|  |  |  |
| --- | --- | --- |
| **Content** | **Totals** | **Obtained** |
| Architecture diagram | 30 | 25 |
| Architecture justification | 20 | 10/0 |
| E/R diagram | 30 | 20 |
| E/R diagram description | 20 | 10 |
| Late submission |  |  |
| **Total** | **100** | **Plagiarism check** |
| **Individual Evaluation** |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**Table of Contents**

[1. Introduction 3](#_Toc87859470)

[2. System Architecture 4](#_Toc87859471)

[2.1 Architecture Diagram—As it is in the prototype code 4](#_Toc87859472)

[2.2 Architecture Diagram—As it should-be 4](#_Toc87859473)

[3. Data Models 5](#_Toc87859474)

[4. Tools and Technologies 6](#_Toc87859475)

[5. Who Did What? 7](#_Toc87859476)

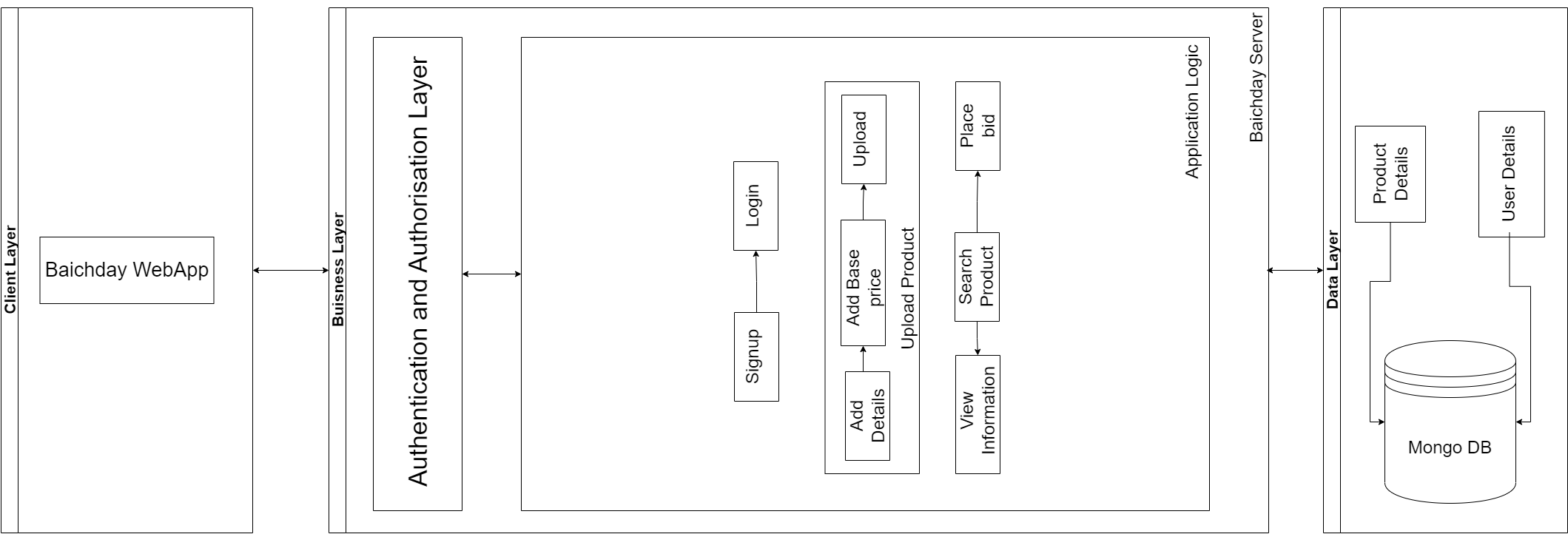
[6. Review checklist 7](#_Toc87859477)

# Introduction

Pakistan is a huge country with a population numbering in the millions, yet the GDP of Pakistan remains rather low as compared to its massive population. We aim to facilitate the people of Pakistan by presenting our auction system. Our software will allow people to auction off their belongings to interested parties. Currently there exists no such platform that supports auctioning as a mechanism. In the status quo, people must spend valuable time and effort searching for customers to purchase their goods at a decent price. We aim to bridge this divide and bring value to the economy of Pakistan by solving this problem. This software will target both businesses and individuals in the Pakistani context. Small scale businesses currently do not have a platform to bid for resources, and shipments. A small scale mobile phone retailer finds it very inconvenient to currently bid for a shipment of mobile phones and similarly, repair shops face the same issue with car spare parts. Our auction system will allow wholesalers to enter their products for retail vendors to bid on. This has the potential to facilitate both wholesalers and the retail industry of Pakistan. Likewise, on a more individual scale, people with valuable assets are unable to put up their goods for the best price and must sell at the highest customer they manage to find. Our model will allow these users to list their belongings and allow bidding on it for the user to find the best price they can get from their belongings. Our software will function similar to an ecommerce marketplace but with the added functionalities of timed biddings, scheduling of bids and other functionalities that will make the auction system a good and worthwhile experience for the users we target.

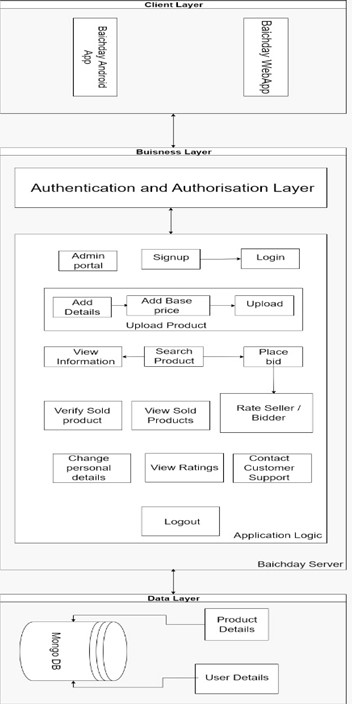
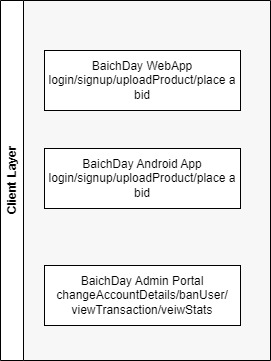
# System Architecture

## Architecture Diagram—As it is in the prototype code



## Architecture Diagram—As it should-be

[You should also add subcomponents in the client layer]



[You content below are in most part similar to content in a document submitted by another group. This appears to be plagiarism.]

Our system is based on two architectural patterns which are the client-server and the model-view-controller.

The client server pattern is a very famous architectural model which is often used for web applications and websites. There are two types of client server models; thin client (logic heavy client) and thick client (logic heavy server). Our software will rely on the thin client server architectural model. This model in our system symbolizes the divide between the front end and backend of our software. The frontend client will be light in terms of logic and will defer to the backend server in terms of computations. On the other hand, the backend will be responsible for most of the heavyweight computations. The backend will connect to the database and retrieve data for the frontend to display.

The frontend on the other hand has the architecture of the Model-View-Controller which is a three-tier architectural pattern. The Model-View-Controller divides the program logic into three interconnected components that are responsible for handling their own responsibilities. Our frontend in this pattern is divided into three portions namely, the model, the view and the controller. The View is the components and the screens that will be shown to the user while the controller will be the segment in contact with the backend. It will receive the data and pass it on to the model for processing. The model will convert it into usable forms for the components of the view to render.

**Maintainability:**

Our three-layered architecture is highly maintainable. That is because each of our layer will be independent of all the other layers in our architecture. Thus, we can update, add, remove or fix our features without really affecting the rest of application as a whole.

[You need to review reusability and extensibility. I would suggest read definitions of reusability and extensibility and then update this section.]

**Reusability:**

Our three-layered architecture a lot of reusability. Since we are implementing our program by writing APIs for each use case, we can easily reuse all of our components when we build an android application as we can just create the front end for it and connect it our already existing business layer so we will not have to create anything from scratch.

**Extensibility:**

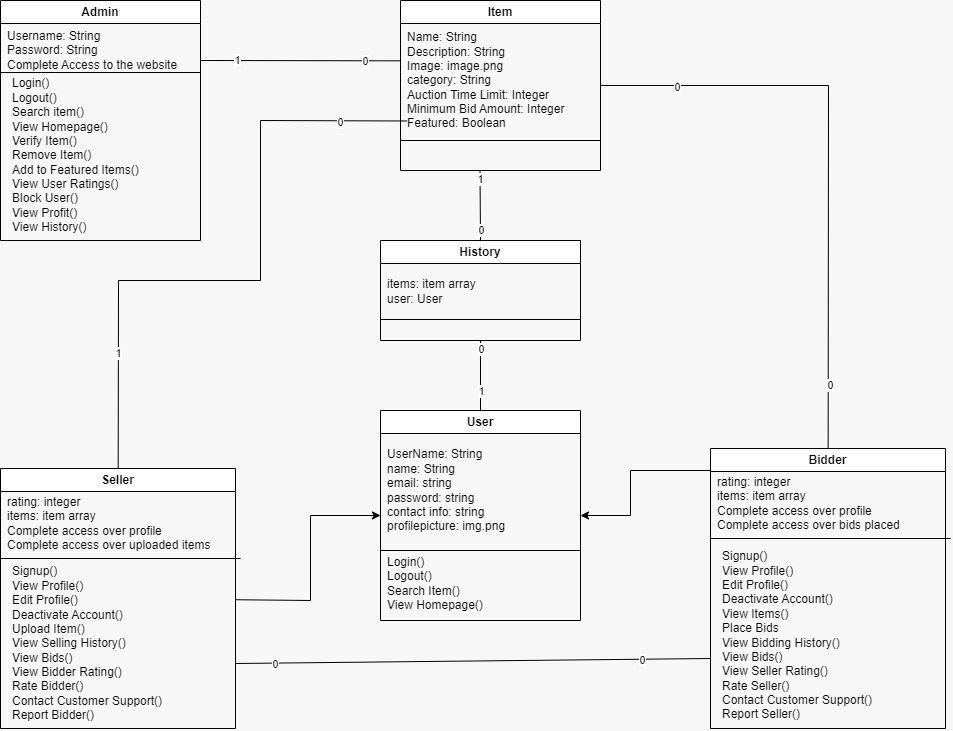
Since we are implementing our application by using APIs for each use case and feature, our model is highly extensible since it will be very easy to add new features and use cases in our existing platform without disturbing or affecting the rest of the application adversely.

**Separation of concerns:**

The layered structure enables the substitution of a complete layer without affecting the rest of the layers as long as the interfaces between the layers stay unchanged. This enables each layer to be scalable independently. When modifications (including changes in interfaces) are made in one layer, only the surrounding layers will need to be altered.

# Data Models

[There should be other entities such as history, different type of users and their access rights etc. ]



**Item:** The item entity contains all the information about the product that has been listed on BaichDay’s website. It contains attributes such as the name of the item, the description of the item, it’s image, the category it falls into and the time limit for the auction along with the minimum bid amount.

**Admin:** Admin will be the superuser that will monitor the issues and other things that need attention on the website. The admin can Ban users, view and remove items and add items to the featured page along with monitor statistics on the website and track profits.

User: On our platform, a user can be a bidder and bid on products that they like, or they can be a seller and upload products for auction on the platform. Users have username, profile picture, name, email and password as their attributes and will have a history of all the products they have placed on bid on or uploaded for auction depending on if they are a bidder or a seller. They will have a rating attribute that will display how good of a bidder or seller a particular person is to the other users. They will have complete access over the products they upload and bid on as well as their information on the website. Users can login, logout and search for items on the platform. They can then place a bid on an item or upload an item based on if they are a bidder or seller respectively.

[Give comprehensive description of each entity.]

# Tools and Technologies

|  |  |  |
| --- | --- | --- |
| **Sr.** | **Tools & Technologies** | **Description** |
| 1. | Visual Studio Code | Visual Studio Code is a simplified code editor used for debugging, code implementation, and version control. It gives developers the tools they require for a fast cycle of code-build-debugging and supports more sophisticated processes and features to the Visual Studio IDE.  [Version 1.72](https://code.visualstudio.com/updates)  <https://code.visualstudio.com/> |
| 2. | Postman | Postman is an API platform for building and using APIs. Postman simplifies each step of the API lifecycle and streamlines collaboration so you can create better APIs—faster.  The Postman platform includes a comprehensive set of tools that help accelerate the API lifecycle—from design, testing, documentation, and mocking to the sharing and discoverability of your APIs  Postman’s full-lifecycle approach to governance lets adopters shift left their development practices, resulting in better-quality APIs, and fostering collaboration between developer teams and API design teams.  [Version 9.4](https://learning.postman.com/docs/getting-started/installation-and-updates/)  <https://www.postman.com/> |
| 3. | MongoDb Cluster | MongoDB is a source-available cross-platform document-oriented database program. Classified as a NoSQL database program, MongoDB uses JSON-like documents with optional schemas. MongoDB is developed by MongoDB Inc. and licensed under the Server Side Public License which is deemed non-free by several distributions.  [MongoDB 6.0](https://www.mongodb.com/new)  <https://www.mongodb.com/home> |
| 4. | React JS | React is a front-end JavaScript toolkit that is free and open-source for creating user interfaces based on UI components. It is kept up-to-date by Meta and a group of independent programmers and businesses.  Version 17.0.2  <https://reactjs.org/> |
| 5. | Heroku | Heroku is a cloud platform as a service supporting several programming languages. One of the first cloud platforms, Heroku has been in development since June 2007, when it supported only the Ruby programming language, but now supports Java, Node.js, Scala, Clojure, Python, PHP, and Go.  <https://www.heroku.com/> |

# 

# Who Did What?

|  |  |
| --- | --- |
| **Name of the Team Member** | **Tasks done** |
| Nashit Iftikhar | Complete document |
|  |  |
|  |  |
|  |  |

# Review checklist

Before submission of this deliverable, the team must perform an internal review. Each team member will review one or more sections of the deliverable.

|  |  |
| --- | --- |
| **Section** **Title** | **Reviewer Name(s)** |
| Arsalan Tarrar | Tools and Technologies |
| Silal Anwar | Architecture |
| Mahad Mubashir | Data Models |
| Moiz Nafay | Introduction |