Project Requirement and Specification on

**Online Auction System**

CSE VI Semester Mini Project

2022-23



**Submitted By**

Name - Shivanshu

University Roll: 2015193

Section: DS and AI

Department on Computer Science and Engineering

GRAPHIC ERA UNIVERSITY, DEHRADUN

# Table of Contents

[**Table of Contents**](#_tobebh21ykm0) **2**

[**About Project**](#_1jf00t76lig) **3**

[**Purpose**](#_k8m45vqlz48x) **3**

[**Abstract**](#_5azpyepgboyn) **4**

[**Requirements**](#_f032vdd4e4m6) **5**

[Hardware Requirements](#_6l6etjslfl8y) 5

[Software Requirements](#_1u20nzqbflh) 5

[**Modules of the Project**](#_sh99x3dymd92) **6**

[Admin](#_dmn5siredxbe) 6

[Seller](#_ln7tu3hzi33z) 6

[Bidders](#_39yj4xi814zr) 6

[Product](#_pspi1ef4zz7x) 6

[**Implementation Details**](#_j6l33addkfu5) **7**

[Testing](#_9j6omauyevgg) 7

[**Database Schema**](#_jjat4aown0wl) **8**

[**Interface Design**](#_1ufnnklc7zoc) **9**

[**References**](#_tafr0vx1abtx) **10**

# 

# About Project

The project “Online auction management system” is a smart management system for conducting auctions online securely and with ease. The project itself has been created on a Python framework called ‘Django’ that provides an easy MVT implementation for creating applications.

The project has been made modular in order to meet the client’s requirements and hence make changes to the code whenever necessary.

# Purpose

The purpose of this project is to build an online auction management system where buyers and sellers come together and trade products. The system consists of a web portal where registered users can add new auctions and place bids in order to buy the items on auction.

Auctions have a name, a description and a photo of the item uploaded by users and a start and end period and minimum price.

The system is realized with 3-tier architecture:

* a relational database that stores the information regarding items, users, auctions and categories of auction
* an application server that cares about the business logic of the system and the presentation layer that consists in the web browser where users can interact with the system.

With such architecture, the database is never directly accessed: for example administrators can change the data stored in the database without connecting directly to it but using their own browser.

# 

# Abstract

An online auction management system that holds online auctions of various categories of products on a website and serves sellers and bidders accordingly.

This system is designed to allow sellers to set up their products auction and bidders to register and bid on the products.

The system consists of following parts:

* **Admin Login** - Where the administrator of the site reads reports and adds categories and bans reported sellers.
* **Seller Login -** Where the seller put their items for auctions.
* **Bidder Login -** Where the bidder bids on the items on auction.

# 

# Requirements

## Hardware Requirements

* Processor - Intel core i3 or higher
* Memory - 2GB or higher
* Hard Disk - 10GB or higher

## Software Requirements

* Operating System - Cross Platform
* Client Side Technologies
  + HTML v5
  + CSS
    - Bootstrap v4.0
  + JavaScript
* Python (v3.4 or higher) Interpreter
* Python Libraries Used
  + Django (v4.0 or higher)
  + Pillow
  + Django-crispy-forms
  + Faker - for testing database with fake data
* Database Server - sqlite

# 

# Modules of the Project

## Admin

* Add or remove categories
* Get Feedback
* Remove reported items
* Ban Reported Sellers

## Seller

* Add Products
* View Products
* Remove Products

## Bidders

* Bid Items
* Report Items
* Report Seller

## Product

* Add Products
* Remove Products

# 

# Implementation Details

Frontend was made using bootstrap and HTML, moreover Django template language (DTL) was used to create dynamic web pages.

For backend, django framework was used, and various models were created which are then mapped by django to the databases.

Class based views are used to perform the CURD (Create, Update, Retrieve, Delete) operations on the models which are created.

To make the application code more modular each module has its own views and separate template directory.

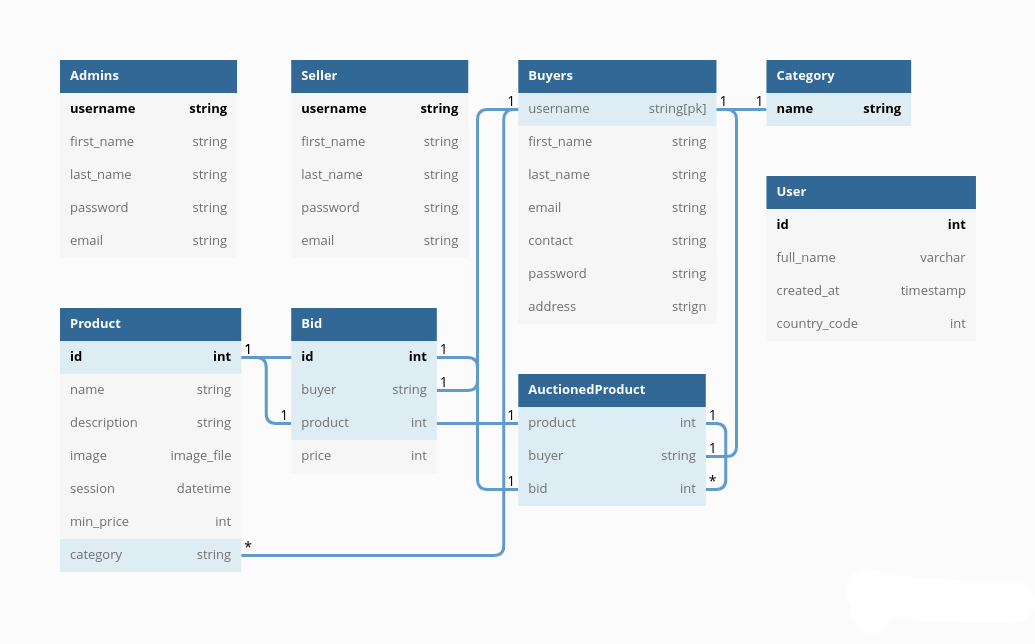
## Testing

The database was populated using the Faker python library. Also python decorators are added to all views to give access to a particular view to a particular type of user.

# 

# Database Schema

Interactive Link - <https://dbdiagram.io/d/62bd631069be0b672c714859>



# 

# 

# Interface Design

# 

### GIthub - <https://github.com/shivanshu-semwal/e_Auction>

# 

# References

* sql docs - <https://app.sqldbm.com/>
* bootstrap v4.6 <https://getbootstrap.com/docs/4.6/getting-started/build-tools/>
* django v.4.0 <https://docs.djangoproject.com/en/4.0/>
* font awesome icons v6.1 <https://fontawesome.com/search>
* style guide for python<https://peps.python.org/pep-0008/>
* frontend mime type for designing forms <https://developer.mozilla.org/en-US/docs/Web/HTML/Element/input/file>