Implementation Document

for

<IITKART>

Version <1.0>

Prepared by

Group: Team 9 **Group Name: Matrix**

Akshunya Vijayvargiya	**200092**	**akshunyavijayvargiya@gmail.cm**
Jasjot Singh	**200468**	**jasjotb02@gmail.com**
Kevalkumar Solanki	**200991**	**kevalsolanki49049@gmail.com**
Kumar Arpit	**200532**	**kumararpit987@gmail.com**
Kushal Gehlot	**200541**	**kushal9427@gmail.com**
Mayank	**200568**	**mayankmayank67936@gmail.com**
Samarpreet Singh	**200848**	**singh.samar23@gmail.com**
Saurav Kumar	**200906**	**sauravsagu@gmail.com**
Subodh Kumar	**201007**	**9703subodh@gmail.com**
Vaidik Sharma	**201079**	**underscoresharma@gmail.com**

Course: **CS253**

Mentor TA: **Swastik Maiti**

Date: **20/03/2022**

Contents

Revisions

Version	Primary Author(s)	Description of Version	Date Completed
Draft Type and Number	Full Name	Information about the revision. This table does not need to be filled in whenever a document is touched, only when the version is being upgraded.	00/00/00

1 Implementation Details

The product IITKART will be deployed as a web application which can be easily accessed using a modern web browser such as Google Chrome, Mozilla Firefox, Microsoft Edge etc..

For the front end, we have used HTML, CSS, and JavaScript for displaying content on the webpage and designing the overall layout. We have also made use of Bootstrap, a CSS framework which helps in adding various components to a webpage and making them responsive.

For the back end, we have used Django – a Python based web development framework that works on the model-view-template (MVT) architecture and the Python programming language.

The database system used in the software is SQLite, a lightweight embedded database that is included directly with Django. This prevents the hassle of installing anything else to support our database or changing the codebase while still providing a robust experience.

We used Bootstrap for some components of the webpage since it provides HTML, CSS and JavaScript based templates that can be customized according to our design specifications.

We used Django for developing much of the web application because of its ability to do more with less code and high scalability. It has many inbuilt functionalities which make the development process significantly faster as compared to some other JavaScript based frameworks.

2 Codebase

Currently, our codebase is present on the following repository:
https://github.com/Akshunya1101/CS253-Team-9

1. All the project files are present in the folder names '*cs253*'.
2. The *accounts* folder contains all the essential python files for running the backend of the website. All the different templates for various pages such as *login, buy, sell, register, home* is present in the *templates/accounts* folder.
3. The *migrations* folder contains some python files storing all the changes made while migrating our database. Majority of the python files are present in this folder.
4. The *models.py* file contains all the models (database tables) for the database of the project, the *views.py* stores all the files for viewing the templates and *urls.py* stores information about URL dispatching.
5. The *crm1* folder contains some predefined python files essential for running the project.
6. The *media/images* folder contains some sample images of the products and is the folder where all the images uploaded by the seller shall be stored.
7. The *static* folder contains static JPEG and SVG files for the website.

3 Completeness

As mentioned in the SRS document, our software can be used as a full-fledged solution for buying and selling used items by the members of a given community and is on par with related solutions like OLX and Quikr.

We enlist the features decided and their implementation status below:

- **The Register page**: We have provided a user-friendly and secure form to collect a unique username, email id, and secure Password based on Django standards. We have deliberately avoided collecting roll numbers to keep the software more general purpose. Also, note that we have not made the Roll number the private key so that users other than students, possibly admins or professors, could keep an account.

- **The Login page (Function F1 in SRS)**: A user can securely log in to his registered account with his Login credentials (User id and Password). One can also navigate easily between the Login and Register pages. When a user logs in, we store a cookie on his browser to remember his session.

- **Footer**: The footer is present on all pages, including the login and register ones. We provide contact details of the site admins and their social handles so that a user may get his issues troubleshot. We also provide access to forms to provide feedback and suggestions and possibly contribute to the software.

- **Access limitation**: Without logging in, a user may not be able to look up details of any item. This ascertains that a third party not belonging to the community cannot meddle with anyone/ anything on the website. Thus, we provide robustness as any ill-activity can easily be traced back to a community member.

- **Homepage/Dashboard (Function F5 and F6 in SRS are implemented on this page)**: Upon logging in, the user is greeted on the dashboard page where one can view the details of the products they have uploaded and quickly navigate to pages to buy/sell the products.

- **Buy Page (Function F3 in SRS is implemented here):** *A user* can see the list of all the items uploaded by other users for purchase.

- **Upload a Product (Function F2 in SRS implemented here)**: Each user can upload an item with its name, image, price and description which will be displayed on the buy page of the website.

- **Search Bar (Function F7 in SRS)**: The search bar allows a user to quickly search up items based on their Names, Descriptions, or Seller to provide a smooth user experience.

- **Chat (Function F4 of SRS implemented here)**: Each product has its separate chat room, which we have provided with the help of an API. This is different from the initially proposed idea but is superior in many ways. It allows for several features such as a fair deal to the buyer, an automatic auction-like system, ease of conveying information to the seller, decreasing the chance of fraud, and maintaining the anonymity of the parties.

- **Filtering and Sorting during Search, and Inventory**: We were unable to implement these features due to a shortage of development time, but these features are not essential and

thus do not break the core functioning of the system. We plan to implement these features in a future version of the software.

The following are the features planned to be implemented and improved in the future versions of the software:

- **Access to only institute members**: Filter to accept only the registrations that have the institute mail id by integrating with an API to send an OTP to the mail id of the account being registered such that only users who have access to an institute email id could register.

- **Updating Footer**: We need to add the social network IDs and create pages/forms for the various options like suggestions/problems/contributions present in the Header.

- **Account page**: We would like add more options to the accounts page such that a user can can manage their privacy settings as well. We will also provide an option for users to delete their accounts if they wish to for privacy reasons or otherwise.

- **Filter and sort options for search**: We would like to improve our search by adding sorting and filtering options by various options such as pricing, seller, item type, etc., since the results might become cumbersome when the database grows large.

- **Inventory**: We would like to provide an option for buyers to wish-list an item so that they may save it for future viewing or to compare it with similar items with ease before making a purchase decision.

- **Chat**: Despite having a common room for a given product, we would still like to have a private chat option between two users on top of it such that they can share their contact details freely there and manage the final payments and stuff. Also, we would like to make the chat more friendly by providing a better frontend interface.

- **Renting**: This was not mentioned concretely in the design document, but we have decided to implement a renting system. Users may temporarily share the items they are not using at the moment with those who need them.

- **Auction**: Again, this was hinted at in the design document but not decided concretely. We wish to implement a dedicated auction system such that parties may get ideal deals for certain popular items.

- **Dark mode**: We would like to enable an option to enable an easily accessible dark mode or light mode for the website based on user preference or their system settings.

- **CAPTCHA**: We would like to enable CAPTCHA to wall a bot or a misintended user while signing from suspected IP addresses.

The total completeness as referred from the functional requirements in SRS is around 90% with few things like renting features, Captcha, and OTP being a few things that we couldn't implement due to time constraints.

Appendix A - Group Log

Zoom meetings:

1) 20th February 2022: General discussion about implementation of the project

2) 27th February 2022: Final discussion about implementation, discussion on resources needed for implementation, and dividing people into respective fields of work.

3) 5th March 2022: meet with TA about our planning and resources and clear some doubt regarding API and some minor functions of the project

4) 7th March 2022: final meet for implementation and discussion about change in the plan due to suggestions of TA
{many offline/online meets in this period for implementation of respective part, solving bugs}

5) 18th March 2022: meet after finishing most of the implementation for finalizing the implementation and checking if we can improve any section of implementation