Implementation Document

for

DealSimplified

Version 1.0

Prepared by

Group 14 **Group Name: Kasukabe Defence Group**

Esra Fatima	**220391**	**esra22@iitk.ac.in**
Kanika Chaturvedi	**220497**	**kanikac22@iitk.ac.in**
Krishiv Geriani	**220545**	**krishivg22@iitk.ac.in**
Manavjeet Singh	**220616**	**manavjeetw22@iitk.ac.in**
Rachit Choudhary	**220845**	**rachitc22@iitk.ac.in**
Riddhima Vijayvargiya	**220883**	**riddhima22@iitk.ac.in**
Ritul	**220896**	**ritul22@iitk.ac.in**
Riya Agarwal	**220899**	**ariya22@iitk.ac.in**
Vishal Singh	**221207**	**vishals22@iitk.ac.in**
Vishap Raj	**221208**	**vishapraj22@iitk.ac.in**

Course: CS253
Mentor TA: Vikrant Chouhan
Date: 28-3-25

Revisions

Version	Primary Author(s)	Description of Version	Date Completed
Version 1.0	Esra Fatima Kanika Chaturvedi Krishiv Geriani Manavjeet Singh Rachit Choudhary Riddhima Vijayvargiya Ritul Riya Agarwal Vishal Singh Vishap Raj	First Draft	27/03/25

1 Implementation Details

1. Programming Languages

- **Python:** The primary programming language used for this project. Python is chosen for its simplicity, readability, and extensive support for web development through frameworks like Django. Its large community and wealth of libraries make it ideal for rapid development.

2. Frameworks

- **Django:** The main web framework used for building the **Lost and Found** and **Marketplace** applications in our project **DealSimplified**. Django is chosen for its **"batteries-included" philosophy**, offering robust features out of the box, such as ORM, authentication, and an admin interface, enabling rapid development and clean, pragmatic design.

- **Django REST Framework (DRF):** Used for building **RESTful APIs**, chosen for its flexibility and powerful features like serialization, authentication, and viewsets, simplifying the process of creating and managing APIs.

3. Libraries

- **django-crispy-forms:** Used for managing and rendering forms more flexibly and elegantly. It integrates with **Bootstrap** for consistent and responsive form styling.

- **crispy-bootstrap4:** An extension of `django-crispy-forms` that provides **Bootstrap 4 support**, ensuring mobile-friendly and visually appealing forms.

- **psycopg2-binary:** A PostgreSQL database adapter for Python, chosen for its reliability and performance in handling database operations efficiently.

- **Pillow:** A Python Imaging Library used for **image processing**, offering ease of use and comprehensive image processing capabilities.

- **dj-database-url:** A utility to configure the **database from a URL**, simplifying database configurations, especially in deployment environments.

- **python-dotenv:** Manages environment variables by loading them from a `.env` file, keeping sensitive information like API keys and database credentials secure.

4. Database Systems

- **PostgreSQL:** The database system used for this project, chosen for its robustness, scalability, and support for advanced data types and performance optimization. It efficiently handles complex queries and large datasets, making it ideal for web applications.

5. Frontend Technologies

- **Bootstrap:** Utilized through `crispy-bootstrap4` for **responsive design and styling**. It provides a comprehensive set of **CSS and JavaScript components**, facilitating responsive and mobile-first web pages.

6. Deployment and Environment Management

- **Environment Variables:** Managed using `python-dotenv` to keep sensitive information (e.g., API keys, database credentials) secure and separate from the codebase.

7. Project Structure

- **Apps:** The project is divided into **two main apps**: `LostNFound` and `Marketplace`, ensuring better organization and separation of concerns.

- **Settings:** Configured to use **environment variables** for sensitive information and support different environments (development, production).

- **URLs:** The URL configuration includes routes for the **admin interface**, **Marketplace**, and **Lost and Found** functionalities.

8. Justification for Tool Choices

The tools and libraries are selected based on their ability to provide a **robust, scalable, and maintainable solution**.

- **Django**, with its comprehensive feature set, enables **rapid development** and easy maintenance.

- The libraries enhance functionality, making the application **secure, performant, and user-friendly**.

- **PostgreSQL** handles complex data operations efficiently, while the modular project structure ensures **clean architecture** and ease of expansion.

Final Verdict: The combination of Django, PostgreSQL, Bootstrap, and environment management practices ensures the project is **scalable, secure, efficient, and well-suited** for both development and production environments.

2 Codebase

Link to the Github Repository :
https://github.com/DrunkenMaster2004/CS253-project.git

1. Project Structure

- **DealSimplified/**: The main project directory containing the core settings and configuration files for the Django project.
    - **settings.py:** Contains the configuration settings for the project, including:
        - Installed apps
        - Middleware
        - Database configurations
        - Static and media file settings
    - **urls.py:** Defines the **URL routing** for the project, linking URL paths to the appropriate views.
    - **wsgi.py:** Provides the **WSGI configuration** for deploying the project.
- **LostNFound/**: This directory contains the application logic for the **Lost and Found** functionality.
    - **models.py:** Defines the database models for:
        - Lost and Found items
        - Categories
        - Claims
        - Images
    - **views.py:** Contains the view functions that handle requests and return responses for the **Lost and Found** application.
    - **urls.py:** Manages the **URL routing** specific to the Lost and Found application.
    - **forms.py:** Contains form definitions for handling **user input** related to Lost and Found items.
- **marketplace/**: This directory contains the application logic for the **Marketplace** application, structured similarly to the LostNFound app.
    - **apps.py:** Configures the **Marketplace application**.
- **Templates:** The project uses **Django's templating system** to render HTML pages.
    - **home.html:** The main homepage template for the **DealSimplified** project.

- **LostNFound/base.html:** Base template for the Lost and Found section.
- **marketplace/base.html:** Base template for the Marketplace section.

---

2. Key Components

- **INSTALLED_APPS:**
  Lists all the Django applications that are part of the project, including:
  - **Third-party apps** like `crispy_forms` and `rest_framework`.
- **MIDDLEWARE:**
  Contains middleware components that process:
  - **Requests and responses**
  - Security
  - Session management
- **DATABASES:**
  Configures the **database settings**, using **PostgreSQL** as the backend.
- **STATIC and MEDIA:**
  Directories for:
  - **Static files**: CSS, JavaScript
  - **Media files**: Uploaded images and other content

---

3. Navigation Tips

- **Start with `urls.py`:**
  - This file provides an overview of the available **routes**.
  - It can guide you to the corresponding **views and templates**.
- **Explore `views.py`:**

    - View functions are the **entry points** for handling requests.
    - They direct you to the relevant models and forms used in the application.

- **Check `models.py`:**
    - Understanding the **data structure** is crucial.
    - Models define the **schema for the database tables**.

- **Templates:**
    - Use templates to understand how **data is presented** to the user.
    - Review how the **frontend is structured**.

Final Verdict:
The **DealSimplified project** follows a modular structure, making it easy to navigate, maintain, and extend.

- Each app is **self-contained** with its own models, views, and templates, ensuring a clear **separation of concerns**.
- The organized directory structure enhances **code readability and maintainability**.

3 Completeness

The "Specific Requirements" section within a Software Requirements Specification Document outlines a comprehensive set of specifications that delineate the desired functionalities of the product.
DealSimplified is a web platform that facilitates buying, selling, and finding lost items. It enables users to upload images of items, predict prices using an integrated ML model, and track lost belongings. The platform ensures user engagement through a robust and user-friendly interface. Our team is continually working to enhance the functionality of our product by introducing new features in the future.

Things that have been implemented :

- **User Dashboard**

  1.User Register
  2.User Login
  3.My Profile
  4.My Whislist
  5.My Chats
  6.User Logout

- **Marketplace**

  1.User Login
  2.User Register
  3.Upload items to sell
  4.Browse Listed Items
  5.Search for Items by Category, Name, or Specification
  6.Predict Estimated price of Items on Sale
  7.Edit, Delete, Update Listed Items
  8.Showing similar Items based on Search
  9.Messaging System for Communication
  10.Notification of Items Added

- **LostNFound**

  1.Report a Lost Item
  2.Report a Found Item
  3.Browse Lost Items
  4.Browse Found Items
  5.Claims to review
  5.Contact Owner/Finder

6.Notification for Lost/Found reported Items
7.Messaging System for Communication
8.My Lost Items
9.My Found Items

Safety Requirements:

- **Mandatory Login**: Users must create a unique ID and password upon first access, ensuring their privacy and security for data they upload (like device images or information).
- **Privacy of User Data**: Any personal information provided by the users (like name, email, phone number, and password) should be kept private and secure. This data must not be shared with any third parties.
- **Data Validation**: Users (sellers) can upload images or details of Items, but the system will ensure that only valid data (like accurate device information) is accepted for prediction.
- **File Upload**: Files uploaded by users must be verified by the system for authenticity and completeness before being stored.

Security Requirements:

- **Confidentiality**: The details of items (such as images, descriptions, or price) uploaded by users should be visible only to authorized users (buyers or sellers) and should not be accessible by unauthorized users.Buyers and sellers should be able to interact through the platform (like messaging or negotiations), but their conversations should not be visible to third party user.
- **Integrity**: The website must ensure that no harm comes to user data or files, ensuring integrity through proper encryption, backup, and validation mechanisms.
- **Availability**: The website and its data must be accessible to authorized users at all times, with minimal downtime.

Software Quality Attributes:

- **Reliability**: The website should function consistently without unexpected crashes, ensuring users' data is always accessible.
- **Performance**: The website should handle multiple user requests efficiently, especially when dealing with high-quality image uploads or predicting device prices.
- **Interoperability**: The website should work seamlessly across different devices and browsers, ensuring a uniform experience for all users.
- **Usability**: The website should be easy to navigate with a user-friendly interface, enabling even non-technical users to upload their devices and retrieve price predictions with minimal effort.

Things we plan to do/update in our web application in future:

Forgot Password Functionality:

- **OTP Verification**: Users (sellers/buyers) who forget their passwords can request an OTP (one-time password) sent via email or SMS. This OTP will allow them to reset their password securely.
- **Benefit**: Enhances security by preventing unauthorized password resets, and provides accessibility for users who forget their credentials.

Mobile App Deployment:

- **Mobile App**: Develop and deploy a corresponding mobile app for iOS and Android to enhance user convenience. This app will allow users to upload devices, track listings, and interact with buyers/sellers more easily.
- **Benefit**: Users can access the platform on the go, providing a more optimized and convenient experience, especially for people who are constantly moving.

Transaction Feature

- **Secure Payment Gateway:** Integrate a secure payment system(e.g. UPI, PayPal) to process transactions between buyers and sellers. Integrate security measures like encryption and two-factor authentication to protect user's payment details.
- **Transaction History:**Allow users (buyers and sellers) to track their past transactions, including the date, device details, amount paid, and status.
- **Notifications:** Send notifications to both parties (buyer and seller) about transaction status updates

Appendix A - Group Log

Date	Time	Attendees	Duration	Minutes of Meeting
08 Jan 2025	18:30	Group Members + TA	60 minutes	• Discussed overall project scope and objectives; • Identified early inefficiencies in environment setup and module separation; • Brainstormed ideas for an ML model to recognize and classify lost items; • Explored potential for a separate ML model to suggest selling prices in the marketplace; • Assigned initial roles
28 Jan 2025	19:00	Group Members (WhatsApp discussion)	30 minutes	• Follow-up on initial design challenges; • Highlighted issues with code merge and database integration; • Brief discussion on potential ML integration approaches for both lost & found and marketplace features; • Reviewed proposed adjustments to the project timeline
15 Feb 2025	19:00	Group Members	55 minutes	• Reviewed progress on module development and encountered complexities in file upload validations; • Discussed inefficiencies in environment variable management; • Evaluated initial ideas for integrating an ML model to enhance lost item identification; • Planned a separate ML model for dynamic price suggestion
28 Feb 2025	19:15	Group Members (WhatsApp discussion)	30 minutes	• Quick status update on resolving merge conflicts and deployment challenges; • Exchanged ideas for improving ML model accuracy in both sections; • Identified a need for additional testing on ML predictions and data validations; • Agreed on drafting future feature enhancements including mobile integration
15 Mar 2025	18:45	Group Members	50 minutes	• Detailed review of challenges: complex database queries, image processing, and code integration issues; • Analyzed potential bottlenecks in ML module integration for lost and found; • Evaluated a prototype of the pricing ML model with initial test data; • Outlined further optimizations for future feature rollouts
25 Mar 2025	19:00	Group Members	60 minutes	• Final review of implementation with emphasis on resolving persistent inefficiencies; • Discussed performance issues related to ML predictions and UI/UX design enhancements; • Mapped out future plans: secure payment integration, mobile

				app deployment, and advanced ML model refinements for both sections; • Summarized overall group efforts