PROJECT REPORT

On

BarterHive

Submitted in partial fulfilment of the requirement for the Course FSE (22CS037) of

COMPUTER SCIENCE AND ENGINEERING
B.E. Batch-2022

in

Jan -2025

Submitted By

Under the Guidance of

Mr. Rahul

Khushi
Roll. No. 2210991795
Kashish Sharma
Roll. No. 2210991770
Kartikay Singh Manhas
Roll. No. 2210991761

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
CHITKARA UNIVERSITY
PUNJAB

CERTIFICATE

This is to be certified that the project entitled “BarterHive” has been submitted for the Bachelor of Computer Science Engineering at Chitkara University, Punjab during the academic semester Jan 2025- June-2025 is a bonafide piece of project work carried out by “Khushi (2210991795), Kashish Sharma (2210991770), Kartikay Singh Manhas(2210991761) towards the partial fulfillment for the award of the course Integrated Project (CS 203) under the guidance of “Mr. Rahul”.

Sign. of Project Guide :

Mr. Rahul

(Trainer & CSE)

CANDIDATE’S DECLARATION

We, **Khushi (2210991795), Kashish Sharma (2210991770), Kartikay Singh Manhas(2210991761),** B.E.-2022 of the Chitkara University, Punjab hereby declare that the Integrated Project Report entitled **“BarterHive”** is an original work and data provided in the study is authentic to the best of our knowledge. This report has not been submitted to any other Institute for the award of any other course.

Sign. of Student 1	**Sign. of Student 2**	**Sign. of Student 3**
Khushi	Kashish Sharma	Kartikay Singh Manhas
ID No 2210991795	ID No 2210991770	ID No 2210991761

Place: Chitkara University

ACKNOWLEDGEMENT

It is our pleasure to be indebted to various people, who directly or indirectly contributed in the development of this work and who influenced my thinking, behavior and acts during the course of study.

We express our sincere gratitude to all for providing me an opportunity to undergo Integrated Project as the part of the curriculum.

We are thankful to “Reshab Kumar” for his support, cooperation, and motivation provided to us during the training for constant inspiration, presence and blessings.

We also extend our sincere appreciation to ***“Mr. Rahul”*** who provided his valuable suggestions and precious time in accomplishing our Integrated project report.

Lastly, We would like to thank the almighty and our parents for their moral support and friends with whom we shared our day-to day experience and received lots of suggestions that improve our quality of work.

Khushi	**Kashish Sharma**	**Kartikay Singh Manhas**
ID No 2210991795	**ID No 2210991770**	**ID No 2210991761**

INDEX

1. Abstract/Keywords

The **BarterHive** is a web-based platform designed to facilitate the exchange of goods and services among community members, promoting sustainable and money-free trading. Utilizing the MERN stack (MongoDB, Express.js, React.js, Node.js), this project enables users to offer items or services they have in excess and search for those they need within their local community. Key features include an advanced barter matching algorithm, real-time chat for negotiation, reputation and review systems, multi-item barter requests, dispute resolution mechanisms, and a community forum for collaborative discussions. The platform aims to foster a sense of community, encourage sustainable consumption, and enable convenient, trustworthy exchanges, all while maintaining secure user profiles and transaction histories. This project not only addresses modern consumption patterns but also creates a localized economy that empowers users to meet their needs through mutual cooperation.

The system goes beyond basic exchange by introducing innovative features such as location-based matching, reputation scores, and dispute resolution mechanisms. Users can engage in multi-item barter requests, ensuring flexibility and convenience in their exchanges. The real-time chat feature allows users to negotiate terms, while the reputation system builds trust by enabling feedback after each transaction. Admin dashboards provide oversight, ensuring smooth and fair transactions within the community.

2. Introduction to the Project

The BarterHive is an innovative web application designed to facilitate the exchange of goods and services within a community without the use of money. Built using the MERN stack (MongoDB, Express.js, React.js, Node.js), the platform enables users to offer items or services they have in surplus and search for those they need, promoting a culture of sharing and sustainability. By focusing on local exchanges, the system aims to reduce waste and encourage efficient resource utilization.

The platform incorporates several unique features to enhance user experience and functionality. With an advanced barter matching algorithm, users are connected based on their needs and offers, while location-based filtering ensures convenience by prioritizing nearby exchanges. Users can negotiate terms via a real-time chat feature and finalize multi-item barter requests, providing flexibility and ease in managing transactions.

A key highlight of the system is its trust-building mechanism, which includes a reputation and review system. This allows users to rate each other after each transaction, ensuring accountability and transparency. In addition, the platform includes a dispute resolution mechanism to handle any issues that may arise during exchanges, along with a community forum to foster engagement and collaboration among users.

2.1 Background

In today’s fast-paced, consumer-driven society, the emphasis on monetary transactions often overshadows traditional methods of resource sharing and exchange. However, barter systems, which date back to ancient civilizations, have proven to be an effective and sustainable way of meeting individual needs. With growing concerns about environmental sustainability and economic inequalities, there is a renewed interest in reviving such systems, particularly within local communities. The idea of exchanging goods and services directly without involving money promotes resource efficiency, minimizes waste, and fosters a sense of collaboration and trust among people.

While traditional barter systems have limitations, such as difficulty in finding matching needs and offers, modern technology presents an opportunity to overcome these challenges. With the advent of web-based platforms, advanced algorithms, and real-time communication tools, the barter process can be digitized, streamlined, and made accessible to a broader audience. A digital barter system eliminates geographical barriers, enhances convenience, and brings transparency to transactions

2.2 Problem Statement

1. Difficulty in Finding Matching Barter Needs

- Traditional barter systems often face challenges in matching users' offers with their needs.
- Lack of an organized platform limits the ability to connect people with complementary goods or services.

2. Lack of Trust and Accountability in Transactions

- Users are hesitant to barter due to concerns about reliability and fairness.
- Absence of a reputation or review system makes it difficult to evaluate trading partners.

3. Limited Communication and Negotiation Tools

- Effective communication is essential for successful bartering, yet traditional systems lack a structured way for users to negotiate terms.
- Without real-time interaction, misunderstandings and incomplete agreements are common**.**

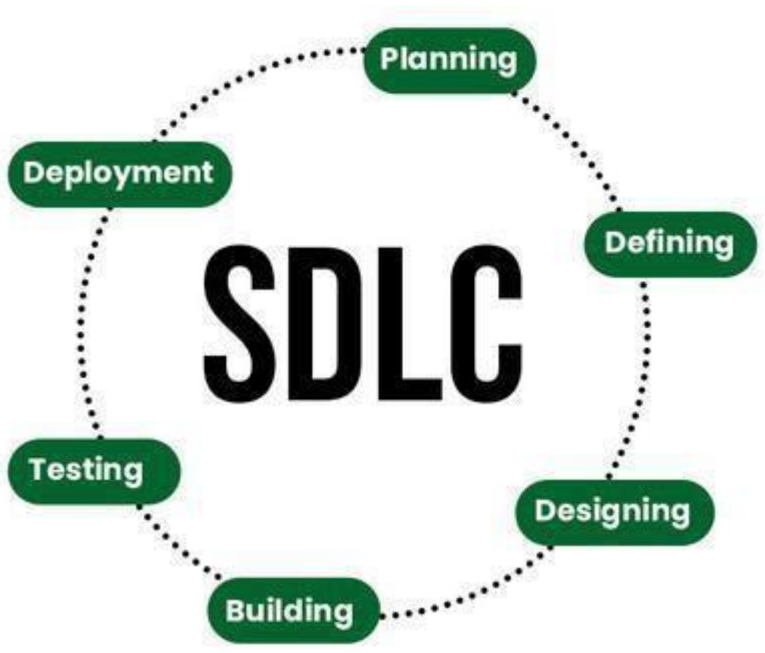
4. Geographical Barriers and Accessibility Issues

- Bartering is often restricted to small groups or specific areas due to logistical constraints.
- Lack of location-based features prevents users from identifying nearby opportunities for exchange.

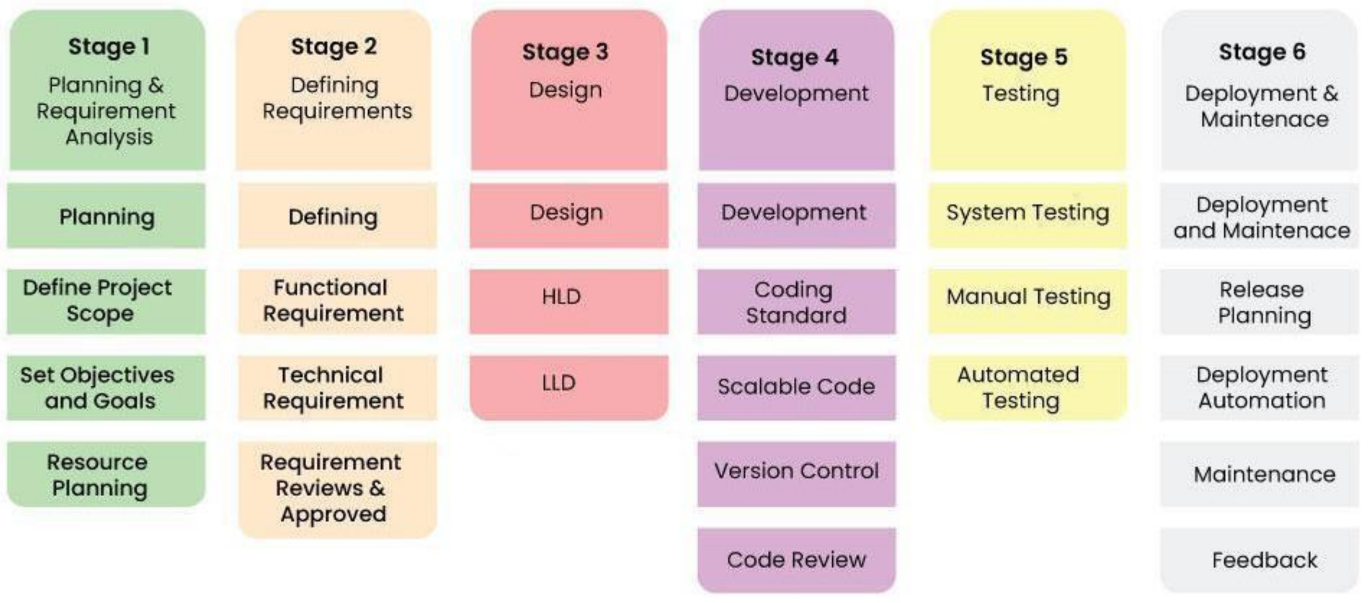
5. Disputes and Lack of Resolution Mechanisms

- Disagreements during or after transactions can lead to dissatisfaction and distrust among users.
- Traditional barter systems lack formal processes to handle disputes, making users hesitant participate.

2.3 Software Development Life Cycle (SDLC)



Software development life cycle (SDLC) is a structured process that is used to design, develop, and test good-quality software. There are 6 stages in this :-



Stage 1 Planning & Requirement Analysis	Stage 2 Defining Requirements	Stage 3 Design	Stage 4 Development	Stage 5 Testing	Stage 6 Deployment & Maintenace
Planning	Defining	Design	Development	System Testing	Deployment and Maintenace
Define Project Scope	Functional Requirement	HLD	Coding Standard	Manual Testing	Release Planning
Set Objectives and Goals	Technical Requirement	LLD	Scalable Code	Automated Testing	Deployment Automation
Resource Planning	Requirement Reviews & Approved		Version Control		Maintenance
			Code Review		Feedback

3. Software and Hardware Requirement Specification

3.1 Software Requirements

Operating System:

- Windows 10/11, macOS, or Linux (Ubuntu recommended for deployment)

Frontend Development:

- React.js: For building the user interface and handling client-side logic.
- Tailwind CSS: For designing a responsive and modern user interface.

Backend Development:

- Node.js: For server-side scripting and managing backend logic.
- Express.js: For building the RESTful API that communicates between the frontend and the backend.

Database:

- MongoDB: For storing patient records, appointment details, hospital information, and other data.

Version Control:

- Git: For version control and collaboration.

Development Environment:

- Visual Studio Code (VS Code): Recommended IDE for coding.
- Node Package Manager (npm) or Yarn: For managing dependencies and packages.
- Postman: For testing API endpoints and backend services.
- Deployment Tools: Netlify

Map Integration:

- Google Maps API : For integrating real-time maps to display hospital locations.

Security:

- JSON Web Tokens (JWT): For secure user authentication and authorization.

3.2 Hardware Requirements

Development Machine:

- Processor: Intel Core i5 or equivalent AMD processor.
- RAM: Minimum 8 GB (16 GB recommended for smoother multitasking and faster builds).
- Storage: At least 256 GB SSD (Solid State Drive) for faster read/write speeds and storage of project files.
- Graphics: Integrated graphics are sufficient, but a dedicated GPU (like NVIDIA or AMD) can be beneficial for handling intensive tasks.

Networking:

- High-Speed Internet: For development, testing, and deployment tasks, ensuring seamless collaboration and deployment.
- Router with Good Bandwidth Support: To handle data transmission during development and testing phases, particularly when dealing with cloud services and real-time APIs.

Backup and Storage Solutions:

- External HDD/SSD: For regular backups of the development environment and database.
- Cloud Storage: Services like AWS S3, Google Drive, or Dropbox for offsite backups and storing large datasets.

4. Program’s Structure Analyzing and GUI Constructing (Project Snapshots)

- **Wireframing:** Create visual representations of the user interface.
- **Prototyping:** Develop interactive prototypes to test the user experience.
- **GUI development:** Implement the user interface using Tailwind and JavaScript(with a framework like React).

The structure of the BarterHive(BH) is critical to ensuring that the application is modular, maintainable, and scalable. The system is developed using the MERN stack, which includes MongoDB, Express.js, React.js, and Node.js. Each component of this stack plays a distinct role in the architecture, working together to deliver a seamless user experience and efficient backend processing.

4.1 Application Architecture

Presentation Layer (Frontend)

- React.js: This is the frontend framework responsible for building the user interface (UI). React.js is used to create reusable UI components, manage user interactions, and handle the state of the application..
- Tailwind CSS: Tailwind CSS is used in conjunction with React to style the application. It allows for rapid UI development with utility-first CSS classes, ensuring the interface is both visually appealing and consistent.

Application Logic Layer (Backend)

- Node.js: Node.js serves as the runtime environment for the server-side application. It handles the execution of JavaScript code on the server, managing client requests and responses.
- Express.js: Express.js is the web application framework used with Node.js to create RESTful APIs. It simplifies the process of routing, middleware integration, and handling HTTP requests.

Data Layer (Database)

- MongoDB: MongoDB is a NoSQL database used to store and manage application data. Its document-oriented structure is particularly well-suited for the complex, hierarchical data used in the BarterHive,

4.2 Component-Level Analysis

Each layer of the BH is composed of several components, each with a specific role within the system. The following section breaks down the key components within each layer:

Frontend Components:

- Navigation Bar: Manages the navigation between different pages of the application .
- Forms: Handles user input for many opeartions.

Backend Components:

- Authentication Module: Manages user authentication and authorization using JWT (JSON Web Tokens) to ensure secure access to the system.
- Middleware Functions: Handles tasks such as error handling, request validation, and logging. Middleware in Express.js is critical for maintaining clean and efficient code.

Database Components:

- Schemas and Models: MongoDB models define the structure of the data stored in each collection, including the data types and validation rules. These models are implemented in the Node.js application using Mongoose.

5 Conclusion

The BarterHive is a modern solution to reinvent the age-old practice of bartering by leveraging technology to overcome traditional limitations. By providing a secure, transparent, and user-friendly platform, it promotes sustainable resource sharing, fosters trust through reputation systems, and encourages community engagement. The inclusion of features such as real-time communication, dispute resolution, and location-based matching ensures that users can trade efficiently and confidently. This project not only addresses economic and environmental concerns but also strengthens social connections, making it a valuable contribution to building self-reliant and sustainable communities.

6 References

- https://www.figma.com/
- https://react.dev/
- https://www.mongodb.com/docs/
- https://expressjs.com/
- https://react-leaflet.js.org/docs/start-introduction/
- https://www.tailwindcss-animated.com/
- https://react-icons.github.io/react-icons/