# A black and red text Description automatically generated

**PROJECT AND TEAM INFORMATION**

## Project Title

|  |
| --- |
| Allergy-Free &Cost Effective Medicine Finder |

## Student/Team Information

|  |  |
| --- | --- |
| Team Name:  Team Code: | CoDevis  DAA-IV-T027 |
| Team member 1 (Team Lead) | Priya – 230223589  [230223589@geu.ac.in](mailto:230223589@geu.ac.in)  priya.jpg |
| Team member 2 | Singhal, Nandini – 23022906  [23022906@geu.ac.in](mailto:23022906@geu.ac.in)  nandini.jpg |
| Team member 3 | Goyal, Sneha – 23022419  [23022419@geu.ac.in](mailto:23022419@geu.ac.in)  **snehaimage.jpg** |
| Team member 4 | Pundir, Urvee – 23022350  [23022350@geu.ac.in](mailto:23022350@geu.ac.in)  urvi.jpg |

**PROJECT PROGRESS DESCRIPTION**

Project Abstract

|  |
| --- |
| Our project is a Allergy-free &Cost Effective Medicine Finder which intends to help users find affordable and allergy-safe medicines by making the user enter name of the expensive medicine. It will provide cheaper alternatives for those medicines having same salts and no allergens.If there are multiple cheaper alternatives then they are sorted by prices. This project aims to provide users pocket friendly medicines while preventing allergic reactions making medicine selection safer and economical for users. |

## Updated Project Approach and Architecture

|  |
| --- |
| Our current approach for the backend system is that it reads medicine data from a CSV file (cleaned\_expensive\_after\_HASHING.csv), which contains columns like expensive medicines and their cheaper alternatives(name, salt composition, price, and unit size).  The program uses C++ with libraries like fstream - file handling, sstream - parsing CSV lines, unordered\_map and vector - efficient data storage, and algorithm - string transformations and sorting. It also employs a priority queue to sort cheaper medicine alternatives based on price.  The program prompts user to enter a medicine name where partial matching is handled. The input is converted to lowercase for case-insensitive matching, then the row of that particular medicine is checked that whether its cheap alternatives are present or not.This was done in a Python program by hashing where matching of salts of expensive with cheap medicines was taking place,the result added new columns in the dataset for easy retrieval. If found, it displays details of the expensive medicine along with cheaper alternatives that have the same salts.Alternatives are parsed from pipe-separated lists in the CSV, and sorted by price using priority queuedue to which we can use min heap and get the cheapest alternative in O(1) time.Users can repeatedly enter different medicines or quit.  This approach ensures quick, user-friendly access to budget-friendly medicine options. |

## 

## Tasks Completed

|  |  |
| --- | --- |
| Task Completed | Team Member |
| 1. Data Preprocessing - All raw medicine data were cleaned and standardized and computing cheaper alternatives by using Hashing and merging two datasets into one 2. Backend for cheap medicine retrieval 3. Frontend (Front page , Login page, Sign-Up page) | Nandini Singhal and Sneha Goyal  Sneha Goyal,Urvee Pundir  Priya and Nandini Singhal |

## Challenges/Roadblocks

|  |
| --- |
| In the backend we faced various challenges –  Problem 1:- Inefficient Grouping of Cheaper Alternative Details  Initially, if a medicine had multiple cheaper alternatives they were displayed incorrectly like first all salts, then all unit sizes, and then prices due to improper comma separation in the dataset.  Solution:- We replaced commas with the ‘|’ inside the CSV columns for alternatives and modified the code to split using |, ensuring correct mapping between salt, unit size, and price.  Problem 2:- Improper and Missing Display for Medicines with Multiple Alternatives  Program only displayed one or no cheaper alternatives even if more were available.  Solution:- We used a priority queue (min-heap) to sort by price and display all cheaper medicines accordingly.  Problem 3:- Case Sensitivity and Partial Matching  Medicine search was case-sensitive and user had to enter full name of the medicine correctly.  Solution:- We applied lowercase conversion (toLower()) to both user input and medicine names, and used find() to allow partial matching.  Problem 4:- Frontend Pages  The main pages of frontend login and signup sections were not visible in center of the page and the background of the pages were not appropriate.  Solution: We created different id for those sections and then implemented in html files.  Problem 5:- Spelling Errors in User Input  The current implementation cannot handle misspelled medicine names.  Planned Solution:- Implement fuzzy string matching to suggest closest valid medicine names.  Problem 6:- Allergy Input not implemented  Our program does not filter medicines based on allergens provided by the user  Planned Solution:- Extend the dataset to include common allergens per medicine and update the program to take user allergy input and exclude medicines containing those allergens during retrieval, ensuring only safe alternatives are shown. |

## Tasks Pending

|  |  |
| --- | --- |
| Task Pending | Team Member (to complete the task) |
| 1. Frontend and backend connectivity 2. Allery dataset preprocessing 3. Backend for allergy manipulation | Nandini Singhal and Priya  Priya and Urvee Pundir  Priya ,Nandini Singhal, Urvee Pundir, Sneha Goyal |

## 

## Project Outcome/Deliverables

|  |
| --- |
| The outcome of this project will be the best alternative with cheapest price for the medicines which have been input by the user.  Also, the user will be able to enter the allergies and the project will not give the medicines that can be allergic to that user. |

# Progress Overview

|  |
| --- |
| The backend logic for searching cheaper alternatives is ready. The frontend pages i.e. home page, login page, signup page  Are designed.  The connectivity of backend with frontend is left.  The user ability to enter the allergy and then filter the medicines according to the allergy is left.  This will be done soon. |

# Codebase

|  |
| --- |
| Repository link:-<https://github.com/SnehaGoyal15/MedFinder>  Branches: main, frontend, backend  backend:-   1. (cheap\_code\_final): Backend for cheap medicine details 2. (cleaned\_expensive\_after\_HASHING.csv): Expensive and cheaper alternatives medicine details grouped by Hashing 3. Colabprograms,files and previous codes: Link to drive which contains- Data preprocessing Python codes in Colab, Initial Data sets ,etc 4. Image:- Output snapshot   frontend:-   1. HTML codes:- index.html, find.html, login.html, signup.html 2. CSS code:- style.css 3. Images:- background images, logo |

## Testing and Validation)

|  |  |  |
| --- | --- | --- |
| Test Type | Status(Pass/Fail) | Notes |
| Backend giving correct answers | Pass | We are getting correct cheaper alternative for the mediciens. |

# Deliverables Progress

|  |
| --- |
| The backend logic for searching cheaper alternatives is ready. The frontend pages i.e. home page, login page, signup page  Are designed.  The connectivity of backend with frontend is left.  The user ability to enter the allergy and then filter the medicines according to the allergy is left.  This will be done soon. |