1. In today’s society, stopping the spread of viruses and diseases is critical. One of the key reasons for this is the increased population density in public malls and retail centres. The global epidemic of Covid-19 necessitates great resilience in these networks. People and employees are exposed to spatial danger if there is no control over crowd density. Graph theory helps in the long-term design of crowd-controlling models that may be used for crowd density estimation, prediction, and control by overcoming the major risk offered by planned shopping times and routes.

Now, we will discuss why we went with this problem.

1. Apps like Google Maps have revolutionised the way we travel and provided invaluable assistance to their users, but there is no such tool on the market that focuses on a more in-depth topic, such as a shopping guide. A user might have several parameters that he or she must consider before heading to a shopping centre or mall, such as the list of items that he has to buy, their priorities, budget, travel distance, availability of a certain product at a given centre, and, especially in the post-covid world, the current crowd density at a particular centre and making sure that the weather is not extreme, which makes the process of determining the most optimum routes, and the order of visiting the shopping malls extremely difficult.

Now, we will discuss what is out ultimate purpose and what are we aiming to do.

1. • Limiting travel and disease spread - By studying information provided by each shopping centre about its availability of items and prices, as well as information about the location of their centres, we can limit the extra and unnecessary travel that we may need to do in search of a specific product, and thus, we can reduce the spread of the virus and diseases. The store pathways and order can be determined ahead of time and given to the shopper.

• Dependent Route Scheduling - The paths and stores visited should be adapt3 able to changing conditions, such as inclement weather or store closing times. In the event of severe weather, a warning should be sent not to leave the house; nevertheless, if the shopper does decide to venture out, the best feasible approach should be to only visit establishments that are as close to the house as possible, allowing for fast assistance.

• Cost Minimization - By optimising the path that should be travelled and minimising any dangers, the shopper can also save a lot of money on transportation. The house and the centres can be thought of as nodes, and knowing the beginning node and the destination allows us to discover the most efficient approach to save time and money.

• Monitoring and Controlling Crowd Density - The one thing that everyone will want in the postcovid world is to keep their hygiene and social-distancing. No one wants to be in a crowded atmosphere, which is why, by analysing data on crowd density in shopping malls, we can recommend malls and shopping centres that are less crowded

1. Now, let us discuss the working of the service. Firstly, we will get the data from the database and we will feed it to the code (network representing the shopping centres and malls). Regardless of the format of the input, it must be saved as a graph (adjacency list representation). The House of the Shopper Node is similarly pre-determined. The distance between this node and all other malls will be calculated and saved for future use. For this, we’ll use Dijkstra’s algorithm. To save time in the future, the shortest path between any two nodes will be determined in advance.

If the weather is terrible, the user will be warned, and it will be suggested that some of the malls on the list that are quite far away from home be removed from the list. The user must next decide whether or not to reduce the overall number of malls to visit. The path-finding algorithm is then invoked, and the proposed path is displayed.

If the shopper’s preference is cost-effective, the shops will be ordered according to how much money they spent there. The path finding algorithm will then be called, and the recommended path will be displayed.

If the customer prefers a less crowded environment, the shops will be classified according to their respective crowd density. The path-finding algorithm is then invoked, and the proposed path is displayed.

If the customer prefers less travelling, we will implement the approximate version of Travelling Salesman Problem using minimum spanning tree. Here, the most optimal solutions for travelling salesman problems are exponential in nature when it comes to time complexities, so instead, we use an approximation algorithm here. Firstly, we remove all the unnecessary edges from the graph using Kruskal’s algorithm for minimum spanning tree, after that, we will visit every edge at most twice only, because in a tree, if we want to traverse the whole tree and return to the starting node, we will only traverse a single edge at most twice.

After that, we will just print the path and return from the function.