Ramdeobaba University, Nagpur

Department of Computer Science and Engineering

Session: 2025-26

Subject: Design and Analysis of Algorithms (DAA) Lab Project

III Semester

LAB PROJECT REPORT

Group Members with Roll number and Section:

Section: A4

Name: Arya Bodkhe

Roll no.:09 **Batch:** B-1

Name: Swati Kundu

Roll no.:48 **Batch:** B-3

GITHUB link of Project repo and deployment link (if website):

https://github.com/Swatikundu21/Smart_Campus_WiFi_Optimizer

TITLE: Smart Campus Wi-Fi Optimization System

Objectives:

- To optimize Wi-Fi router placement across a smart campus environment.
- To ensure efficient signal coverage within a fixed budget using algorithmic optimization.
- To apply multiple algorithmic strategies (Knapsack, Sorting, and MST) in a real-world scenario.

- To minimize total cabling distance and cost while maximizing coverage and capacity.
- To develop a professional, interactive dashboard for visualizing optimized results.

Introduction

The **Smart Campus Wi-Fi Optimizer** project aims to automate and optimize the placement of Wi-Fi access points (APs) across a campus environment. With the growing need for reliable connectivity, it becomes essential to design a network layout that provides maximum coverage with minimal cost and effort.

This system uses data-driven decision-making to identify optimal router positions considering parameters like signal strength, user capacity, cost, and estimated user density. The optimization process applies three key algorithmic techniques — **Sorting**, **Knapsack**, and **Minimum Spanning Tree (MST)** — for efficient network design. The final output visualizes the selected access points and their interconnections using a professional Streamlit dashboard.

Algorithms/Technique used:

1)Sorting algorithm

Purpose:
To prioritize Wi-Fi router locations based on a computed score that combines signal strength, user capacity, and cost-efficiency.

Algorithm Steps:

1. Compute a score for each location using the formula:
   `score = (signal_strength × 0.5) + (capacity × 0.3) – (cost × 0.002) + (estimated_users × 0.2)`
2. Sort all potential router locations in descending order of their score.
3. Select top-scoring candidates for further optimization in the Knapsack algorithm.

```
Algorithm Sort_Router_Locations
Input: List of routers with attributes (signal_strength, capacity, cost, estimated_users)
Output: Routers sorted in descending order of score

1. For each router in list:
       score = (0.5 × signal_strength) + (0.3 × capacity)
             - (0.002 × cost) + (0.2 × estimated_users)
2. Sort the routers based on score in descending order
3. Return the sorted list
End Algorithm
```

Explanation:

Sorting ensures that high-performing router locations are evaluated first, improving the overall optimization process and reducing unnecessary computation.

2) 0/1 Knapsack Algorithm

Purpose:

To select the most beneficial Wi-Fi router locations under a limited budget constraint.

Algorithm Steps:

1. Initialize a dynamic programming (DP) table of size `(n+1) × (budget+1)` where n is the number of routers.
2. For each router `i`:
    a. Include it if its cost ≤ remaining budget.
    b. Compare the total score between including or excluding the router.
3. Continue until the table is filled.
4. Trace back through the DP table to find the selected routers.

```
Algorithm Knapsack_Optimization
Input: Routers[] (each with cost and score), Total_Budget
Output: Selected subset of routers with maximum total score within budget

1. n ← number of routers
2. Create DP table of size (n + 1) x (Budget + 1)
3. For i ← 1 to n:
       For w ← 0 to Budget:
           If cost[i] ≤ w then
               DP[i][w] ← max(DP[i-1][w], DP[i-1][w - cost[i]] + score[i])
           Else
               DP[i][w] ← DP[i-1][w]
4. Initialize Selected ← []
5. w ← Budget
6. For i ← n down to 1:
       If DP[i][w] ≠ DP[i-1][w]:
           Add router[i] to Selected
           w ← w - cost[i]
7. Return Selected
End Algorithm
```

Explanation:

This algorithm ensures that the maximum total signal quality and user capacity are achieved without exceeding the given budget.

Diagram (conceptual):

Budget vs. Cost Matrix (DP Table) where selected cells represent optimal router combinations.

3)Minimum Spanning Tree (Prim's Algorithm)

Purpose:

To minimize the total cable distance required to connect to the selected Wi-Fi access points.

Algorithm Steps:

1. Start from any selected node (router).
2. Find the shortest edge connecting it to an unselected node.
3. Add that edge to the MST and mark the new node as selected.

4. Repeat until all nodes are connected.

```
Algorithm Prim_MST
Input: Coordinates of selected routers
Output: Set of edges forming the Minimum Spanning Tree and total distance

1. Compute distance_matrix between all router pairs
2. Initialize selected[0] ← True
3. total_distance ← 0
4. For count ← 1 to n-1:
       min_distance ← ∞
       For i ← 0 to n-1:
           If selected[i] = True:
               For j ← 0 to n-1:
                   If selected[j] = False and distance_matrix[i][j] < min_distance:
                       min_distance ← distance_matrix[i][j]
                       x ← i, y ← j
       Mark selected[y] ← True
       Add edge (x, y) to MST
       total_distance ← total_distance + min_distance
5. Return MST, total_distance
End Algorithm
```

Explanation:
The MST ensures all routers are connected with the least possible cabling cost, improving network efficiency.

Time Complexity and its explanation:

Algorithm	Time Complexity	Explanation
Sorting Algorithm	**O(n log n)**	The sorting algorithm arranges all possible Wi-Fi router locations based on their computed score (signal strength, capacity, cost, and estimated users). It ensures that the most suitable locations are considered first during optimization.
0/1 Knapsack Algorithm	**O(n × W)**	The Dynamic Programming approach to the Knapsack problem iteratively fills a DP table of size `(n+1) × (W+1)`, where `n` is the number of items (routers) and `W` is the maximum budget. This ensures optimal router selection under cost constraints.

Minimum Spanning Tree (Prim's Algorithm)	**$O(V^2)$**	Prim's algorithm connects all selected routers with the minimum possible cabling distance. For each vertex, it repeatedly selects the shortest edge connecting to an unvisited vertex, resulting in a total time complexity of $O(V^2)$.

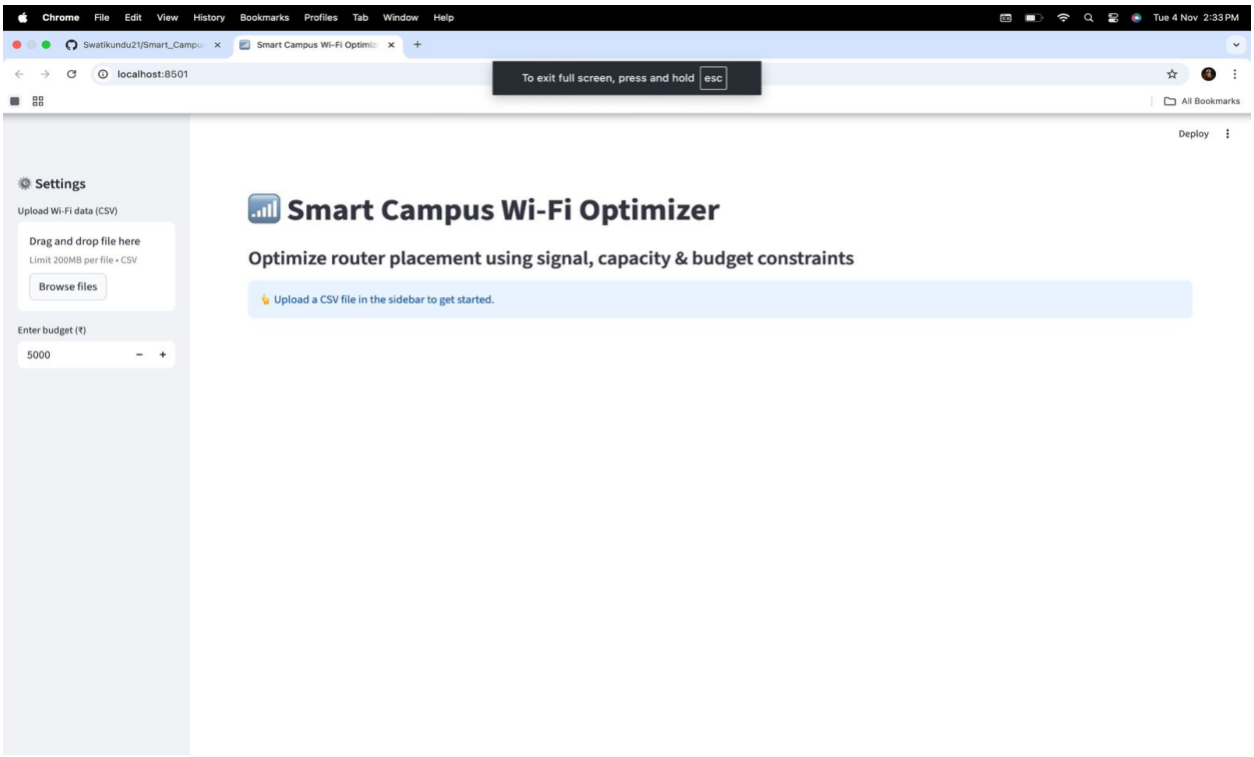
Results and Outcomes

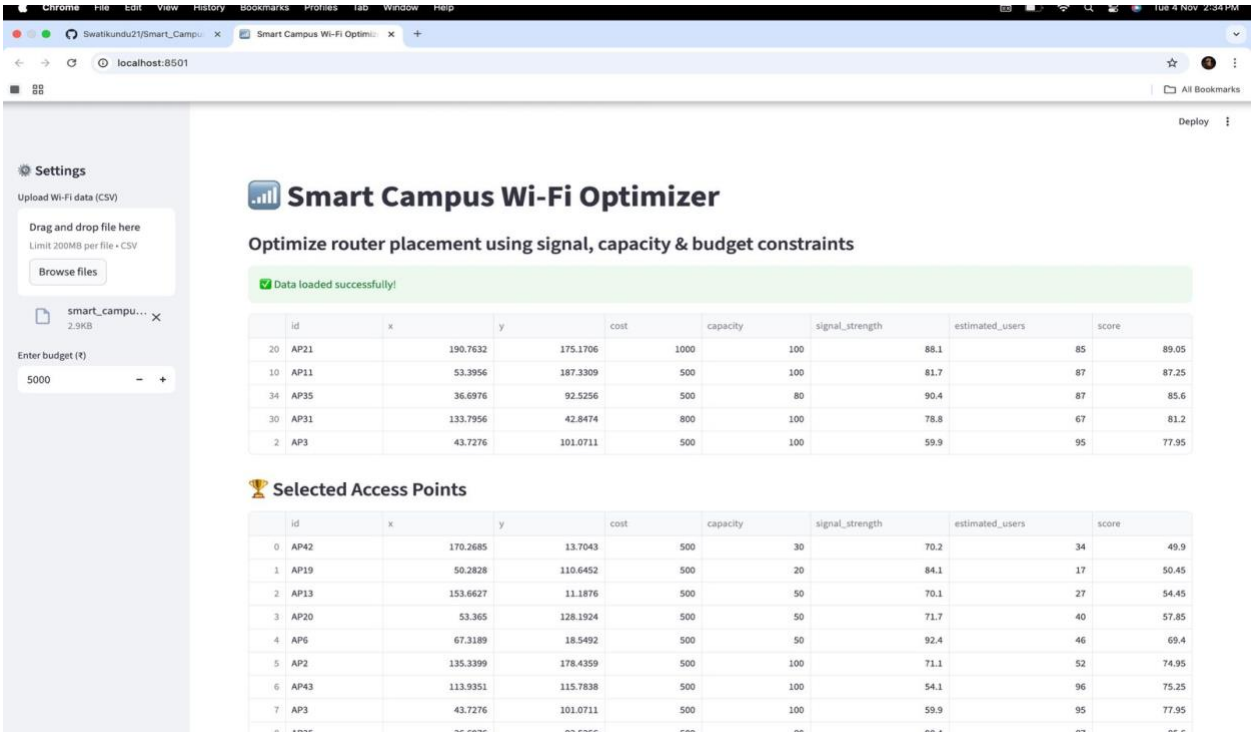
The Smart Campus Wi-Fi Optimizer was executed successfully using the Streamlit dashboard. The user uploaded a CSV file containing Wi-Fi location data and specified a budget value. After uploading, the system calculated a performance score for each access point based on signal strength, capacity, estimated users, and cost.

The optimized results displayed two tables — one showing all access points with their computed scores, and another showing the **selected optimal routers** within the given budget using the **Knapsack Algorithm**.

The **Minimum Spanning Tree (MST)** was then applied to connect these selected routers with minimum total cable distance. The visualization clearly showed all router positions (grey points), selected routers (blue points), and their MST connections (red dashed lines). The total cabling distance obtained was approximately 411.17 units for default budget of 5000.

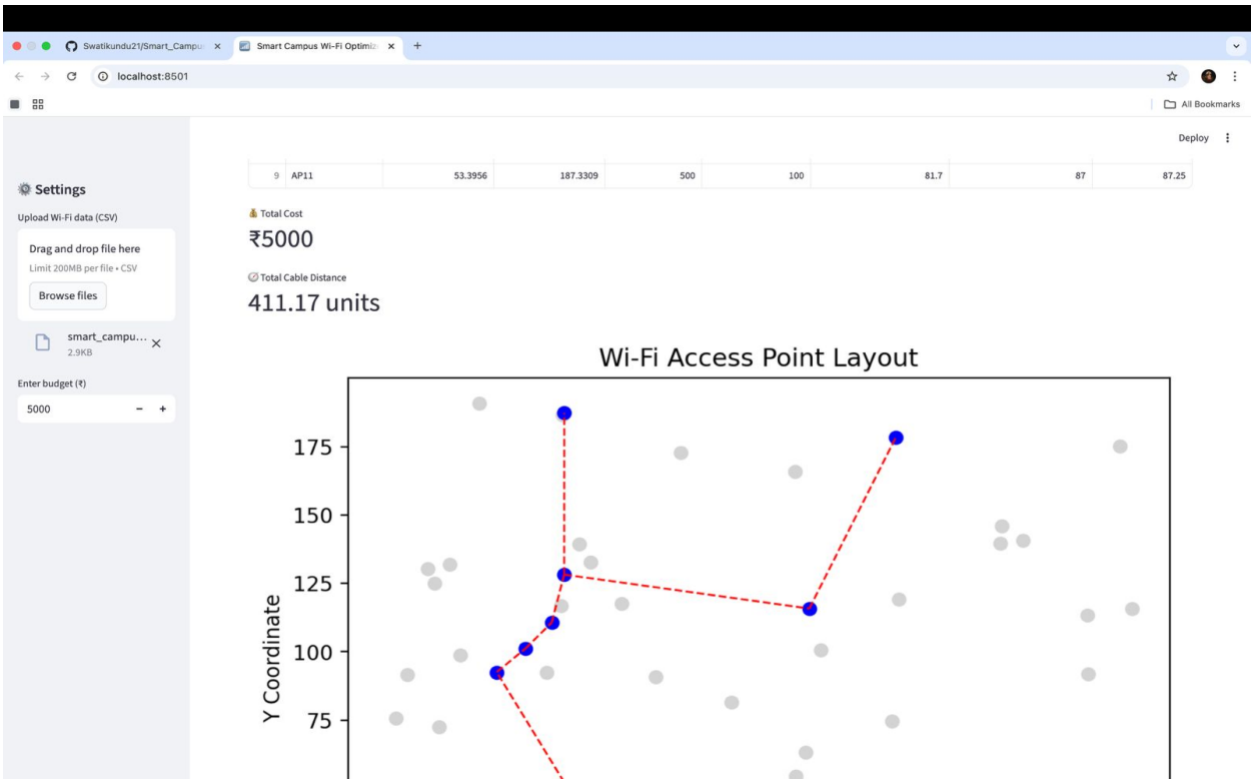
Overall, the system successfully identified cost-effective and well-connected router placements. The final dashboard also displayed total cost, total distance, and provided an option to download the optimized router configuration as a CSV file.

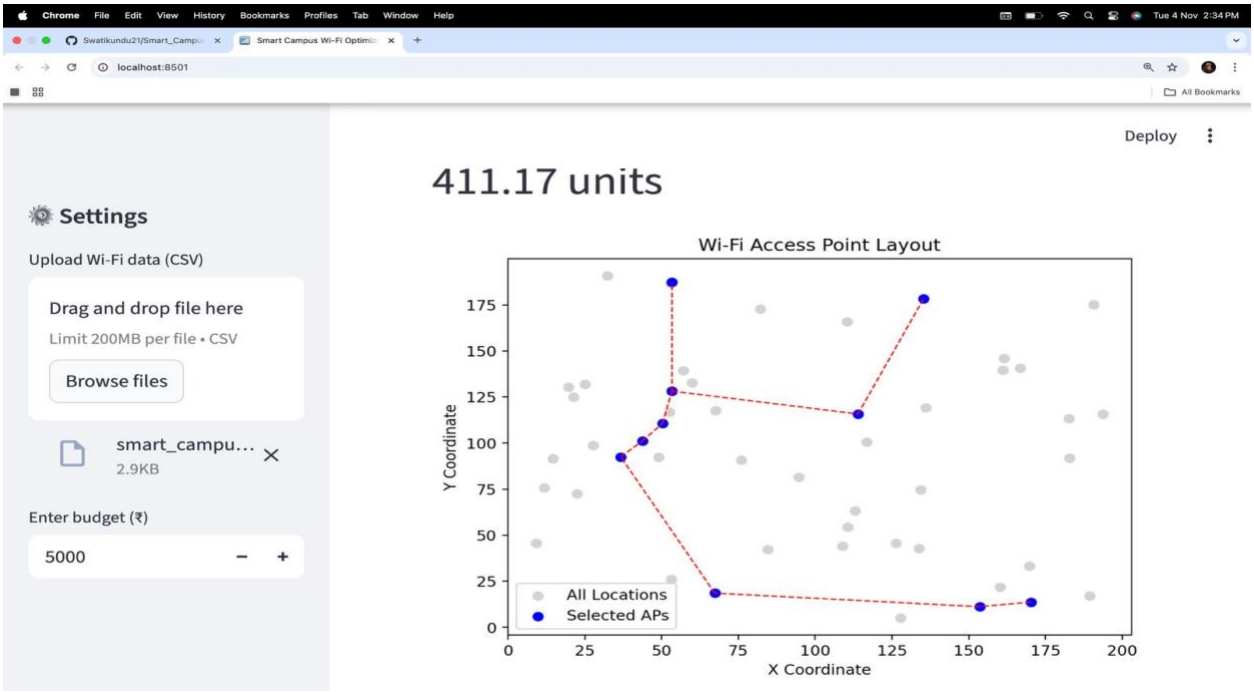


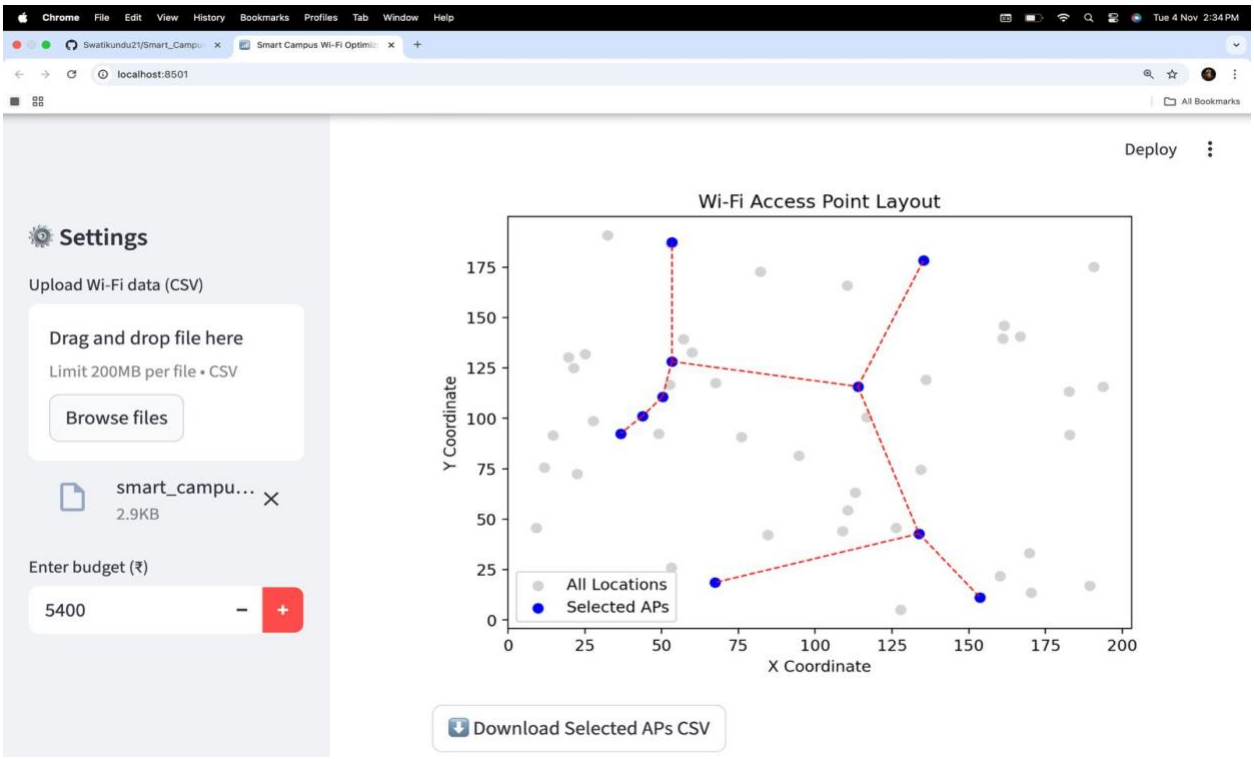


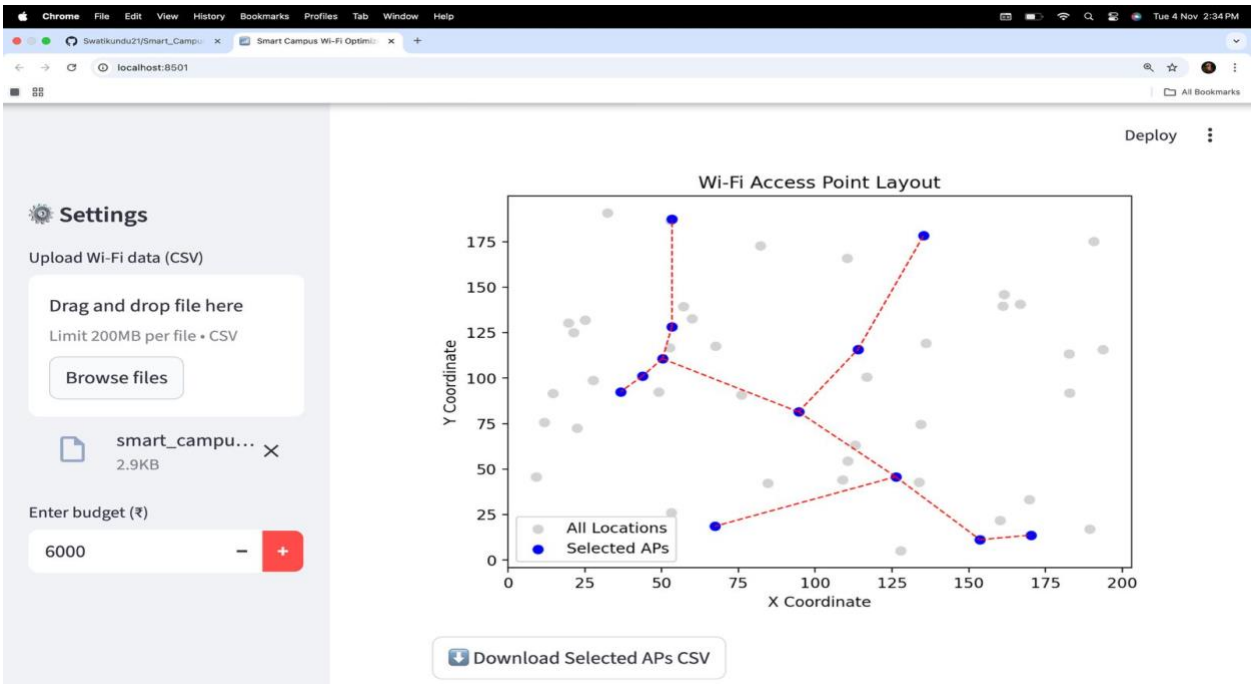
	id	x	y	cost	capacity	signal_strength	estimated_users	score
20	AP21	190.7632	175.1706	1000	100	88.1	85	89.05
10	AP11	53.3956	187.3309	500	100	81.7	87	87.25
34	AP35	36.6976	92.5256	500	80	90.4	87	85.6
30	AP31	133.7956	42.8474	800	100	78.8	67	81.2
2	AP3	43.7276	101.0711	500	100	59.9	95	77.95

	id	x	y	cost	capacity	signal_strength	estimated_users	score
0	AP42	170.2685	13.7043	500	30	70.2	34	49.9
1	AP19	50.2828	110.6452	500	20	84.1	17	50.45
2	AP13	153.6627	11.1876	500	50	70.1	27	54.45
3	AP20	53.365	128.1924	500	50	71.7	40	57.85
4	AP6	67.3189	18.5492	500	50	92.4	46	69.4
5	AP2	135.3399	178.4359	500	100	71.1	52	74.95
6	AP43	113.9351	115.7838	500	100	54.1	96	75.25
7	AP3	43.7276	101.0711	500	100	59.9	95	77.95


Smart Campus Wi-Fi Optimiz
localhost:8501
All Bookmarks
Deploy
Settings
Upload Wi-Fi data (CSV)
Drag and drop file here
Limit 200MB per file • CSV
Browse files
smart_campu…
2.9KB
Enter budget (₹)
5000
9 AP11 53.3956 187.3309 500 100 81.7 87 87.25
Total Cost
₹5000
Total Cable Distance
411.17 units
Wi-Fi Access Point Layout
Y Coordinate
175
150
125
100
75


Chrome File Edit View History Bookmarks Profiles Tab Window Help
Tue 4 Nov 2:34 PM
Smart Campus Wi-Fi Optimiz
localhost:8501
All Bookmarks
Deploy
411.17 units
Settings
Upload Wi-Fi data (CSV)
Drag and drop file here
Limit 200MB per file • CSV
Browse files
smart_campu…
2.9KB
Enter budget (₹)
5000
Wi-Fi Access Point Layout
Y Coordinate
X Coordinate
All Locations
Selected APs
0
25
50
75
100
125
150
175
200





Conclusion and Future Scope

Conclusion:

The Smart Campus Wi-Fi Optimizer effectively demonstrates the real-world application of algorithmic techniques in network planning. By combining **Sorting**, **Knapsack**, and **MST**, it

delivers an optimal configuration within given constraints. The interactive dashboard enhances user understanding and decision-making.

Future Scope:

- Integration with live IoT data for dynamic signal updates.
- Implementation of heuristic or AI-based optimization (e.g., Genetic Algorithms).
- Expansion for multi-campus or large-scale city networks.
- Incorporation of real-time cost and energy efficiency metrics.
- Cloud-based deployment for cross-device accessibility.