Summary
Currently creating an easy-to-use software solution to create schedules for businesses based on predicted labor using Python fundamentals alongside SQL, and the Tkinter library for the GUI.

Outline:

Input Gathering:
Gather input regarding:
Total number of employees
Total hours needed to be covered
Expected hours per shift
Employee availability (if available)

Data Preparation:
Calculate the total number of shifts needed (total_shifts = total_hours_needed / expected_hours_per_shift).
Prepare a list of employees and their respective availability (if available).

Schedule Generation:
Implement a scheduling algorithm to distribute shifts among employees. Some common algorithms include:
Greedy algorithm: Assign shifts to available employees starting from the one with the least assigned shifts.
Genetic algorithm: Generate schedules by simulating evolution over multiple generations, with fitness determined by factors like shift preferences, availability, and fairness.
Constraint satisfaction algorithm: Ensure that each employee's constraints (availability, maximum hours per week, etc.) are satisfied while generating the schedule.
Assign shifts to employees based on the algorithm chosen.
Output:

Generate a schedule report:
Display each employee's assigned shifts.
Display any unfilled shifts or conflicts.

Optional Features:
Shift preferences: Allow employees to specify preferred shift times or days off.
Fairness considerations: Ensure fair distribution of undesirable shifts (e.g., night shifts) among employees.
Schedule validation: Check if the generated schedule meets labor laws and company policies regarding breaks, maximum working hours, etc.
GUI or web interface: Create a user-friendly interface for inputting data and viewing schedules.

Testing and Refinement:
Test the program with various scenarios and edge cases to ensure correctness and robustness.
Refine the scheduling algorithm based on feedback and performance.

Deployment:
Once tested and refined, deploy the program for regular use.