User Story 1: Loading and Displaying the City Map

As a dispatcher, I want to load a city map into the application and see it visually displayed, so that I can understand the layout of the city, including intersections, streets, and the location of the warehouse for planning delivery tours. The map should clearly show all key elements, such as streets connecting intersections and the warehouse as the starting point for all deliveries. If there are any issues with loading the map, I want to be informed of the problem and have the option to try again or load a different map, ensuring I can proceed with my work without unnecessary delays.

User Story 2: Managing Couriers, Delivery Requests, and Modifications

As a dispatcher, I want to input and modify delivery requests, so that I can handle delivery tasks effectively and adapt to changes. By default, there should be one courier, but I should be able to adjust the number of couriers based on workload. For each delivery request, I want to specify or load a pickup point (represented as a square) and a delivery point (represented as a circle) from the map. The system must validate data accuracy and ensure that each pickup is visited before the corresponding delivery.

I also want to interactively modify the delivery program by adding or removing delivery requests. Each modification should trigger a recalculation of the schedule and update the tour, including arrival and departure times, in real-time. If a courier cannot fulfill a task due to time constraints, I want to be notified.

User Story 3: Optimizing and Displaying Delivery Tours

As a dispatcher, I want the application to compute and display optimized delivery tours for all couriers based on the program of pickup and delivery requests, so that routes are efficient and respect all constraints. The optimization must ensure that the overall length of the tour is minimized while adhering to constraints such as visiting each pickup point before its corresponding delivery point, respecting delivery time windows to ensure deliveries occur within the specified time limits. Additionally, the application must support multiple couriers by addressing the Vehicle Routing Problem (VRP), optimizing routes collectively across all couriers rather than individually. The calculations must also account for the fixed courier speed of 15 km/h and a stop duration proper to each location. Once optimized, the tours should be visually represented on the map, with each courier's route displayed distinctly and annotated with details such as addresses, arrival times, and departure times.

In scenarios where no feasible tour can be generated due to these constraints, the system must notify the dispatcher. Any modifications to the delivery program, such as adding or changing requests, must trigger dynamic recalculations to ensure the updated tours continue to respect all constraints while maintaining efficiency.

User Story 4: Saving, Restoring, and Error Handling

As a dispatcher, I want to save the current delivery tours to a file and restore them later, so that I can back up my work and resume planning without starting over. The saved file should include all relevant details, such as the map, courier information, pickup and delivery requests, and calculated routes with timing details. When restoring, the application must correctly reload the tours and display them on the map. If an error occurs during saving or loading, the system must provide a clear message and allow me to retry or take corrective action.