Challenge: Widgets

Create a basic system description and document a normalized schema from the attached widgets text file. Include 1) what you think this system would do 2) what you feel would be a reasonable database structure for the data and a reasonable architecture for the system 3) any questions or concerns you have regarding this dataset/system that might need to be answered before establishing an ideal database/solution for such a system.
It's a very open-ended problem, and that's part of the problem.

Answer:
There are mixed data in only one table, which is not readable and programmatically nor correct for saving information as table in database. You should normilize database by segmenting for Suppliers, Customers, Warehouses and Widgets tables and designate each with ID for relationship between derived tables. For example supplier_id, customer_id, warehouse_id and widget_id. And sort data form table by these ids. This is method is “One-to-One relationship” or “One-to-Many” which is reasonable for services (it is point of as developers).

Table #1 Suppliers

Supplier_id	Supplier
sid_1	Little Traps
sid_2	Big Traps
sid_3	Raytheon

Table #2 Customers

Customer_id	Customer
cid_1	Home place
cid_2	Bug Store
cid_3	No Bears R Us

Table #3 Warehouses

Warehouse_id	Warehouse
wid_1	AUS

wid_2	ATL
wid_3	MSP

You can see below normilized tables like Widgets and Packages of Customers. I select to save the quantities and costs in table Widgets and a price and packaging in talbe Packages of Customers. The relationship between two entities will realize by widget_id. Probably, Widgets table could be useful for ordering new Widgets and tracking inventory.

Table #4 Widgets

Widget_id	Widget	Supplier_id	Warehouse_id	Qty	Min_qty	Cost
wtid_1	Ant Trap	sid_1	wid_1	112	50	0.5
wtid_2	Mouse Trap	sid_1	wid_2	200	50	1
wtid_3	Bear Trap	sid_2	wid_3	10	10	40
wtid_4	Moose Trap	sid_2	wid_3	5	5	50
wtid_5	Elephant Trap	sid_3	wid_3	3	5	90

Table #5 Packages of Customers

Package_id	Customer_id	Widget_id	Packaging	Price
pid_1	cid_2	wtid_1	bag of 10	10
pid_2	cid_2	wtid_1	bag of 5	6
pid_3	cid_2	wtid_2	bag of 5	15
pid_4	cid_1	wtid_1	bag of 10	9
pid_5	cid_1	wtid_1	bag of 5	5
pid_6	cid_1	wtid_2	box of 2	5
pid_7	cid_1	wtid_2	box of 1	3

pid_8	cid_1	wtid_2	bag of 10	20
pid_9	cid_1	wtid_3	box of 1	50
pid_10	cid_1	wtid_3	box of 5	220
pid_11	cid_1	wtid_4	box of 1	75
pid_12	cid_1	wtid_5	crate of 1	100
pid_13	cid_3	wtid_3	box of 1	60
pid_14	cid_3	wtid_4	box of 1	80
pid_15	cid_3	wtid_5	crate of 1	110

In this table, we can see that different Customers are paying different prices for the same Widgets in the same *Packaging*. (Of course, I am assuming for now that I correctly identified a relationship to exist among the entities Widget, Packaging, and Price). For instance, cid_2 (*Home Place*) gets a *10-count bag* of wtid_1 (*Ant Traps*)for $1 less than cid_2 (*Bug Store*).

1) This database could be part of a system that supports a small Sales Interface. I imagine this interface could help help Sales Representatives at a small Widget Manufacturer/Distributor to create specially priced Packages of Customer for their Retailers.
2) As developer I think normilize table as one-to-one relationship will be reasonable a reasonable architecture for the system. Perhaps, it will use indexing fields in a table for optimize request to database where probably will use one-to-many relationship for normalize tables in the future.
3) Regarding the system I would want to know exactly what kind of system is desired and how it meets our business needs. Do we need to track inventory? Automate invoicing? Something else?