Systems Analysis and Design Challenge

Local Agriculture Networking System

Network connectors are applications allowing two groups of people to connect through a common portal. For instance, Uber and Lyft connect individuals needing transportation with other individuals willing to offer their time and transportation to those individuals. Prior to these network connectors, such a synergy was not readily available.

Small agricultural entities find similar difficulty in peddling their wares. While large agricultural entities have more secure marketing channels for their goods, small local farmers cannot often as easily find outlets to sell their goods in a timely manner.

A non-profit organization that promotes locally sourced, organic produce and other farm-to-table goods has asked you to design a flexible application attracting local buyers and providers, facilitating those connections for both the consumer and vendor. They will handle the marketing of this app to jumpstart its usage. After conducting interviews with a group of stakeholders, including small farmers, community members, farm markets, and larger buyers, you summarized the following system requirements:

- A small agricultural business must apply to market their wares via this application.
- The seller must register each item offered, including produce type, amount available within a specific timeframe, price, and designation of each item as organic, naturally grown, fair trade, or other commonly recognized certification. (Though produce is the first item of interest, the app should be built in such a way that would support other items such as dairy and meat.)
- The seller can also designate one or more preferred methods of selling in their profile: including CSA (community-supported agriculture, in which individual households pay to receive "shares" of agriculture), farmer's market booths, and larger local store purchases of mass quantities of produce.

- The name and location of each seller will be visible, but specifics regarding the business (such as seller's personal name, address, e-mail, and other identifying characteristics) will only be available to system administrators for privacy purposes.
- The system administrator must approve the small business's application.
- Buyers can view small business offerings without registering. They should be able to see "new" additions on a "trending" display, and drill down for details related to seller produce being offered.
- If a buyer wishes to connect with a seller, they must register as either a business or individual potential buyer.
- The buyer must also be approved by the system administrator.
- The buyer can use the system to post a need for the purchase of a specific item (and quantity, at a specific price, in a specific timeframe).
- The buyer can use the system to find available produce and make offers. An individual buyer could accept the offer of a weekly CSA share, or a farm market could invite the seller to fill a booth. A local Meijer store could attempt to source a pallet of locally grown apples.
- The system must allow both buyers and sellers to rate their transaction(s). Consistently low ratings would cause the administrator to deactivate a seller and/or a buyer from use of the system.
- Sellers and buyers can "opt in" or out regarding receiving notifications for marketing purposes. All communications within the app will occur through a "blind" notification system to hide this information. The system must be designed to support "push" marketing, in which sellers and/or buyers are contacted with new offerings that might appeal to them, based on historical activity.

YOUR ASIGNMENT IS to use only one technique (either Object Oriented OR Structured/Traditional Technique) to specify how the system should operate. If you use a structured technique, you must specify the flow of data inside the system. If you use an OO technique, you must specify the classes inside the system and how they are used in order to achieve the system's objectives.

WHAT TO TURN IN: If you are using the structured/traditional approach, you are expected to turn in the following:

1. A Context Diagram.
2. A level 0 (zero) Dataflow Diagram.
3. A Level 1 DFD for each one of the processes that you identified in your Level 0 System DFD.
4. Process descriptions for the processes contained in your DFDs.
5. An Entity Relationship Diagram (ERD) showing the 3rd Normal Form Database that will support the system you designed.
6. Prototype with Windows Forms and/or Web Pages.

If you are using an Object-Oriented approach then you are expected to turn-in the following:

1. Use-case Diagrams.
2. Use-case Descriptions.
3. Sequence and/or Activity Diagrams.
4. A Class Diagram (for objects in persistent storage).
5. State machine diagrams.
6. Prototype with Windows Forms and/or Web Pages.

For creating models, use your own business modeling software. This could include any Visio, CASE, ICASE or other model-based development product.

The prototype must be developed based on your models. It does not have to be fully implemented; however, a system design that provide mocked up screens with window form/web page interaction will be considered in the overall grading. The screens can be created using any graphical drawing software (such as Microsoft Paint or Photoshop), wireframing tool, or you can take screen shots from development tools (such as Microsoft Visual Studio, Access or Eclipse). Given the time limit of the contest, handwritten mock-ups are allowed; however, the screens created by computer software will be given better grades.

Ensure that your team number is written on every sheet that you turn in. If you write your name or school anywhere, your team will be disqualified. Number all the sheets that you turn in, sequence them and account for them, i.e.: Page 3 of 7, Page 4 of 7, Page 7 of 7, etc.

Contest Evaluation

The judges will use the following categories in evaluating your team solution. The models that your team is required to develop depend on which methodology is selected.

NOTE: Competitors are expected to utilize ONE and ONLY ONE Analysis and Design approach. Using a combination of components from both the Structured/Information Engineering approach and the Object Oriented approach should be avoided.

	%	**Structured/Information Engineering**	**Object-Oriented Approach**
Information Flow	**40**	Decomposition, DFDs, Dependency and Process Action Diagrams	Use Cases, Sequence, and/or Activity Diagrams
Information Structure	**40**	Entity Relationship Diagrams (ERDs) and Data Constraints	Class Diagrams (for objects in persistent storage) and State Charts (State Machine Diagrams)
Prototyping	**20**	Windows, Screens, and/or Web Pages (Prototypes/Wireframes)	Windows, Screens, and/or Web Pages (Prototypes/Wireframes)
Overall	**100**		