Binary Ninjaz Prize Justification

Innovation

The Harvest system is targeted at people who are not the most computer literate. This has meant that our system must be very simple to use for anyone who picks it up. However, as people become accustomed to the system and the interface we would like to provide them with more power. To do this we've employed progressive disclosure into the user interface design of the system.

Our app on the surface is vary simplistic, but once you start *needing* certain things parts of the UI will hopefully lead to solving your problem. For example, when using the search functionality in web, you may start of by doing a simple search for a foreman's name, but later you think to search for a session that has 2 workers. Both of these will! Once that works you might start trying to search for a session that has a worker called "John" or "Peter" or any other interesting things.

To achieve giving the user a progressively more powerful search functionality we implemented our own domain specific language (DSL). We give the user a lot of power without having them learn a complicated system.

Some examples of search functionality:

- "peter": shows all items that have some field that contains the word peter.
- "peter or john": shows all items that have some field that contains the word peter or john.
- "peter john": shows all items that have some field that contains the word peter and some field (other or the same as peter) that contains the word john.
- "foreman peter": shows all items with a foreman that is named peter.
- "worker john sum": shows all items with a worker called john. Also shows the sum of the yield collected.
- "last 3 days orchard name block a": shows all sessions in the last 3 days that had work done in the orchard named block a.

Algorithmic Innovation

While developing our system we encountered challenges that required using certain algorithms. We would like to discuss 3 design challenges that required us to turn to certain algorithms.

Definitions:

- Yield: The amount of produce collected on a farm.
- Session: A work period where produce is collected.
- Orchard: An area where workers will collect produce in.
- Pickup: A location and time associated with a worker collecting a bag.

Genetic Algorithm

One of the requirements for our system was to provide expected yields for farmers. This proved to be challenging for a couple reasons. We did not have much historical data of previous yields to base our models on if we were to use something like a neural network. Secondly, There are many different considerations to take into account, factors such as farm size, crop and seasonal factors, which would mean we would need an adaptable solution.

To overcome these issues we decided to use a basic sinusoidal regression. This simple formula gave us a sine graph that covered the range of yield amounts and gave us fairly decent results. However, these results were far from satisfactory. So we turned to using a genetic algorithm to help improve the accuracy of the prediction. Due to our choice in database we were left to implement our genetic algorithm in JavaScript with a service that gave us limited CPU time. This meant we had to ensure that our algorithm was as efficient as possible to ensure we reduce server costs.

Convex Hull

For a farmer to see how orchards are performing orchards need to be mapped out on a map in the form of a polygon. It was found that this was not always the easiest task for people to complete. So instead we decided we should build an inferred orchard area from the points of sessions that selected a certain orchard.

To solve this we turned to using a convex hull algorithm namely the Graham scan algorithm. As each session is completed we take the points of pickup and union them with the set of points that currently make up an orchard. We apply Graham scan to this new set to form the new orchard area. But we only do this if an orchard area is not manually specified by a farmer.

Ray Casting

To reduce the chances of breaking referential integrity in our system we decided to make points not tied to anything other than a session. This meant that determining where pickups are made must be calculated. This problem essentially broke down to if a polygon contains a point.

We employed a ray casting algorithm to determine if a point is located in a certain orchard or not.

Triple Bottom Line

The aim of Harvest is to provide an easily accessible and cheap solution for farms of all sizes. While there are farms that have advanced tracking and monitoring of their workers and crops these are usually large industry farms that can easily afford paying extensive costs.

The Harvest system is built to use as little resources as possible. We only rely on mobile GPS systems which are now very commonly found in phones. Along with an internet connection there's nothing else you need to start using the Harvest system.

As price is a concern we needed to minimise internet costs. To do this we employed caching on the mobile systems. Our use of a realtime database meant we could have more control over
downloads and uploads. Since the database supported notifications we only download data for
a user when data is updated on the server once.

Continuing on the note of reducing costs we employed subsystems in Harvest to help prevent theft. Theft is a major concern for many farmers who cannot always easily monitor their workers.
The Harvest system has realtime location tracking of foremen during a working session. We provide
stats to help see both under performing workers and areas. For instance you can look at a heat map
and see where an area is not doing so well and see that might be area were workers are stealing from.

To be less cynical stats can also be used to see if certain areas of the land just aren't producing as much because of a pest or any other environmental reasons.