Team 7 FoodSaver App

Software Requirements

User requirements –

1. FoodSaver should authenticate login information of vendors

2. FoodSaver should display a list of stores nearby that have food items for sale for buyers

3. FoodSaver should display a list of items that a specific vendor has on sale

4. FoodSaver can create and store a list of pinned items that are on sale from a chosen vendor.

5. FoodSaver can filter/sort out items specified by category, price, distance, and/or expiration.

6. Vendors can delete items on FoodSaver that they are no longer selling or have expired.

System requirements –

1. User request vendor’s location, Food Saver sends query to database for addresses of vendors and matches with user specified vendor. Returns vendor’s location to user
2. User requests a category of food items from a vendor, Food Saver queries database for items that match specified category. Returns a list of items from available vendors.
3. On the date a food item is set to expire, the item is hidden from buyer view and highlighted in red text to vendors to remind them to remove the item

Risk Analysis

Developing this app, the most sensitive thing that we need to be aware and cautious is the security of the personal information of the users. Most of the general problems that can affect the functionality of the app don’t represent a big risk at all.

Ex- Information such as username, email address, and password of user is leaked, or easily reachable.

Risk – Losing user, and being vulnerable to hackers displaying false information in the app.

We will use the Firebase Authentication System which is secure, and the risk of this happening should be low.

Ex- After release users found bugs/crash

Risk- User would dislike the app and would engage them to delete and not use the app

To prevent this, we would do SMOKE testing before release for 2 environments. And if this happens, we will evaluate the priority and severity of the task, if a bug is critical, add to the current Sprint, roll-out according to roll-out plan. push hotfix if it’s possible to back-end or web.

Ex - Bad feedback about the product

Risk- Clients would not download and use the app because they dislike it.

To prevent this, we could do testing before the release to fix critical bugs, a soft launch to find and fix bugs + receive user feedback early. And if this happens, we can answer user feedback, fix problem, reward a person, who found an issue (if a problem is in some issue) and make marketing out of this

Ex- Changes in scope

Risk- Project scope is not reasonably feasible for time allotted  
To prevent this downgrade scope of project to be manageable between the few weeks allotted.