SOFTWARE DESIGN DOCUMENT (SDD) FOR

320 Green Team Project

Version 2.0
December 1, 2015

Prepared for:
Sunderland/Leverett, MA Health Inspector

Prepared by:
Ather Akhtar, Andrew Chang, Peter Marathas, Andrew Marchetti,
Michael Markman, Eric Maryea, Neven Recchia,
Shawn Sowersby, Alex Sullivan, and Josh Tranfaglia.
University of Massachusetts
Amherst, MA 01003

320 GREEN TEAM PROJECT: HEALTH-E
TABLE OF CONTENTS

Section **Page**

1 INTRODUCTION

The purpose of this software design document is to provide a low-level description of Health-e, providing insight into the structure and design of each component. In short, this document is meant to equip the reader with a concrete understanding of the inner workings of the Health-e system.

1.1 GOALS & OBJECTIVES

The purpose of Health-e is to replace the current physical method of archiving Board of Health data from restaurant inspections, well reports, and septic tank reports with a new, digital system. The intended user of the tablet aspect of this product is a health inspector who will be using the tablet to document or browse for data while on sight at an inspection. Accordingly, the final product must be quick, efficient, and easy to use. It must offer the appropriate features to the user without overwhelming them or requiring them to spend valuable time learning how to use said features. The user interface should be intuitive and easy to learn so that the system acts as a tool for the user, not something that slows them down while they are doing their job.

1.1.1 Browsing and Search

The user must be able to quickly and efficiently find the data that they are searching for on the tablet. This requires a streamlined browsing and search interface that allows the user to visualize results in an easy-to-read manner. They will be able to enter a search query into a search bar that will always exist in the same location on the page, and results will be displayed in a list format with the most relevant results at the top. The focus here is that the user should not have to spend time searching through the list of results to find the data they are looking for; maneuvering through the results should be a quick, intuitive process.

1.1.2 Form Entry

Form entries should model the paper version of the forms in which the inspector would normally fill out on location. They will not replicate the paper forms exactly since that would cause the user to have to constantly zoom in and out on the tablet to fill in sections of the form due to the size of a typical tablet screen. Rather, all fields that appear on the paper forms will appear on the tablet forms in a simple, clean format. The user interface here should be predictable and exist in the natural, chronological order in which the inspector would typically fill out the fields so that it is efficient and easy to use.

1.1.3 Delayed Upload

The user should not have to have a second thought about delayed upload. That being said, the system must effectively notify the user of the current status of certain documents. Having an icon system that identifies documents on the tablet as being uploaded or in queue for delayed upload will allow the user to remain in the loop as far as the status of their documents in a simple, noninvasive manner.

1.2 PROJECT OVERVIEW & SCOPE

The Health-e system is comprised of two main user components: a web app and a tablet app. For the sake of this design document, we will only be focusing on the tablet app portion of this system. Within this document, the terms user and health inspector will be interchangeable as they are one in the same for this project. The following are some definitions of common phrases that will be used in this document.

Term or Acronym	Definition
UML	Unified Modeling Language
DFD	Data Flow Diagram
SDD	Software Design Document, aka SDS, Software Design Specification
SRS	Software Requirements Specification

1.2.1 Core features

The tablet app will include the following core features:

1. User Authentication
   - Asks the user for their unique password to ensure that the information stored on the tablet app is safe and secure
2. Form Entry
   - Streamlines the form entry process by providing all necessary fields in a clean, organized format
   - Provides restaurant, well, and septic forms available to choose from with ease
   - Replicates the original forms in terms of color coding, field names, and information
3. Location Entry
   - Documents the current location of the user in terms of coordinates
   - Maps the user's location and their proximity to other septic tanks and wells in the area
4. Browsing and Search
   - Allows the user to complete a search query using the search bar
   - Displays search results in a scrolling list form that is easy to maneuver through
   - Displays search results in a variety of orders depending on the user's specification
5. Form Printing
   - Displays what a form will appear as on a printed page with options to print or cancel
6. Delayed Upload
   - Categorizes completed forms as either uploaded or pending for delayed upload
   - Displays to the user which forms fall in which category based on icons associated with the forms

1.2.2 Addtional features

Below are some features that are not guaranteed to be incorporated in the final product, but could potentially be included, time permitting. Due to their tentative nature, they will not be covered in this document.

1. Accessibility
   - The user could be able to change certain accessibility options such as form layout, background customization, and font size or style
2. Device Synchronization
   - The user could be able to synchronize data across multiple devices such as tablets and mobile devices

1.3 SOFTWARE CONTEXT

Health-e will ideally be made available on both the Android and iOS market free of charge. Development and maintenance costs are essentially nonexistent, so funding will not be an issue. The final product will exist under the confines of this course and future extensions to the product, while possible, will probably not be taken up by the group currently implementing this product, if at all.

Of the core features mentioned above, only the *Form Entry*, *Browsing and Search*, and *Delayed Upload* features will be discussed further in this document as those are the only features that the Green Team will be focusing on for this project.

1.4 MAJOR CONSTRAINTS

The main constraint for the Health-e project is time. There is approximately a month allocated to the development, testing, and release of this product. This is not a sufficient amount of time to properly produce a fully functional product, so some aspects of the development will probably be neglected. While all of the core features for the tablet will be included in the final product, as well all of the features not mentioned in this document for the web app and database, the functionality of these features may not be compromised.

1.5 INTENDED AUDIENCE

While the SRS document is intended for a more general audience, the purpose of this design document is to layout a visual template and functional flow of the final product and is thus intended for individuals who may be working directly on this product. This includes software engineers, project consultants, and team managers.

1.6 REFERENCES

Food Inspection Form: https://people.cs.umass.edu/ ridgway/cmpsci320/customer/FoodInspectionForm.pdf
Septic Pumping Report can be found under "Title 5 Official Inspection Form" via this link:

http://www.mass.gov/eea/agencies/massdep/water/approvals/title-5-septic-system-forms.html

2 ARCHITECTURAL AND COMPONENT-LEVEL DESIGN

2.1 SYSTEM STRUCTURE

Health-e will be written in HTML5, CSS and Javascript. We will use the Ionic Framework to deploy the application for iOS and Android simultaneously. Ionic utilizes AngularJS and Sass to offer libraries for mobile-optimized HTML, CSS, and Javascript components. Additionally, we will be using the Ionic Material library for an advanced UI for our app.

2.1.1 Justification

The selection of the Ionic Framework followed the consideration of other frameworks including the native iOS and Android SDKs. After recognizing that a health inspector should not be limited to any single device, a priority was placed on the availability of the tablet application.

The Ionic framework builds off of Apache Cordova and allows for the development of a single code-base that creates an Android, Windows, and iOS application. The framework is an extremely popular open source product with a large community to help with any problems that arise. Ionic is also designed with the native SDKs in mind allowing users to have a native iOS, Android, or Windows experience.

2.1.2 UML Diagrams

Data Entry/Browsing and Search:
Figure 1. shows the UML diagram used for both the browsing and search functionality as well as the data entry. Each form type contains a list of inspections and are sub-types of an abstract ReportType. This structure of data modeling was used to best represent shared data the all three report types contain while still allowing each of them to have multiple inspections. The User class, an instance of which is pulled upon login, is used by the Restaurant class to indicate the person performing the inspections. This is in alternative to having a separate attribute in the Restaurant class for the inspector which would be duplicating stored data. Also, of the three report types, only Septic uses an instance of the Location class in addition to the one inherited, so that the owner's location can be stored if it is different from the actual location of the well.

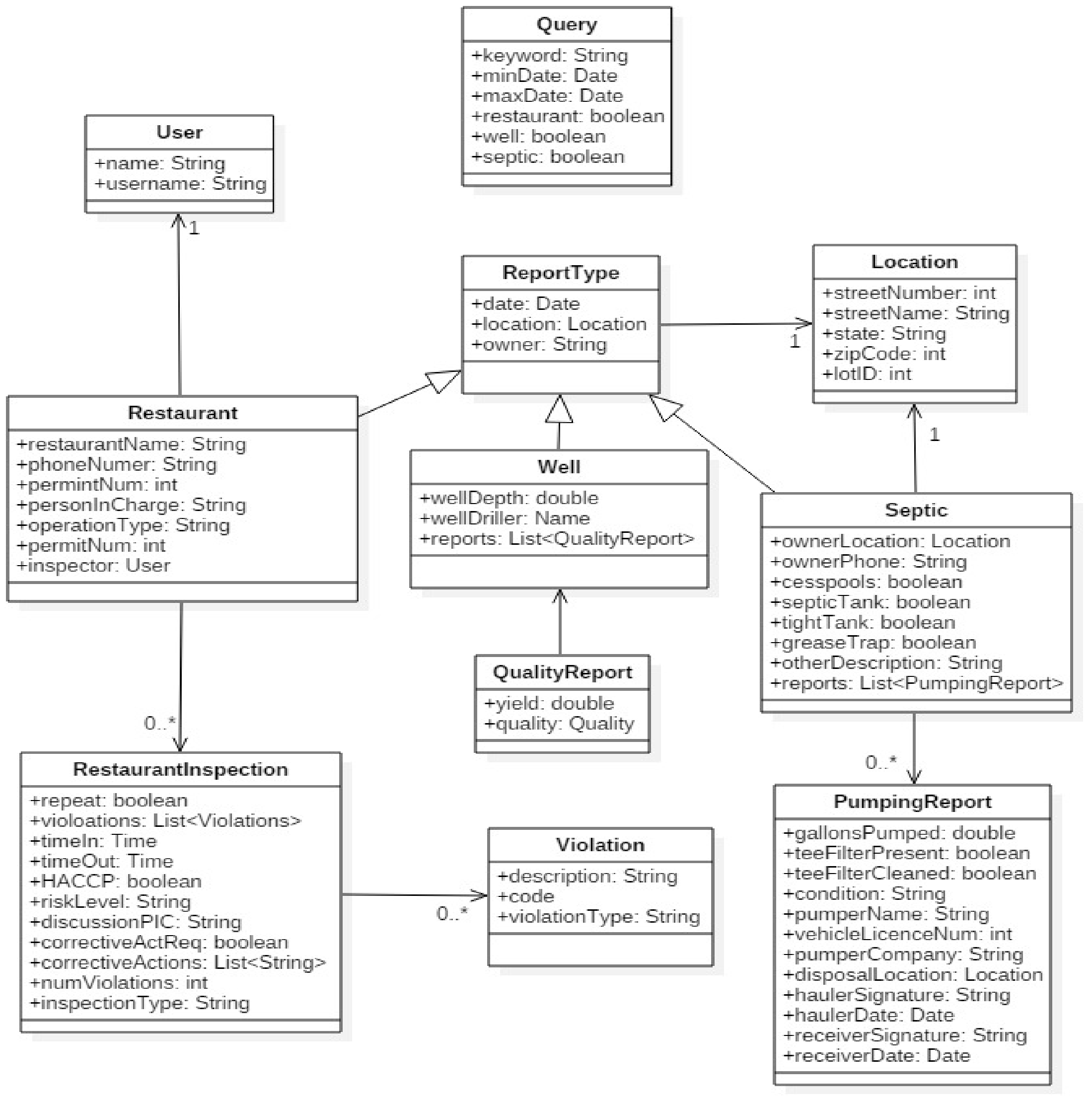


Figure 1: Abstractions for data on forms and search.

Delayed Upload:
When control is passed to the DelayedUploadController a few processes can be run. The following Figure 2 illustrates when an update is run locally.

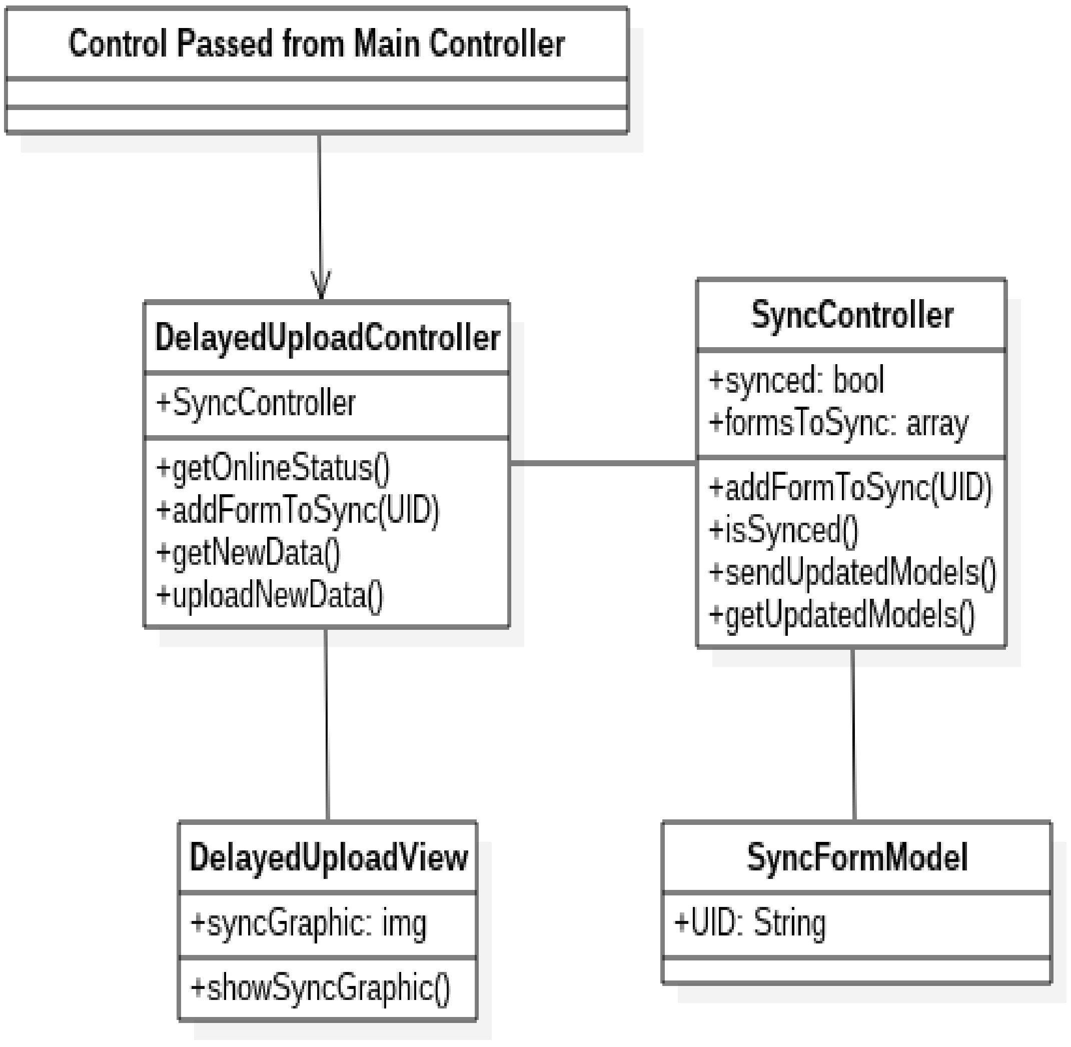


Figure 2: Abstractions for the system's uploading of forms.

2.2 SYSTEM IMPLEMENTATION

All classes and functions described in this section will be written in Javascript. All classes are meant to represent and contain all relevant information pertaining to their real life incarnations. Thus, class attributes for reports are predominantly taken from official documents, and all supporting classes utilize as much real-world information as is relevant.

2.2.1 Class Definitions

- restaurant - contains basic information on the restaurant
- report - Superclass for restaurantReport, wellReport and SepticReport classes. Contains most basic report/form information.
- restaurantReport - Subclass of report. contains information on restaurants that pertain to the restaurant inspections.
- wellReport - Subclass of report. Contains all fields pertaining to Well Report input.
- septicReport - Subclass of report. Contains all fields pertaining to Septic Report input.
- location - Specific information pertaining on where to find a specific location .
- user - References the user/inspector that is logged into the system and submitting a restaurant report.
- violation - Specifics to the type of restaurant violation including a text description.
- query - Contains ability to search for past submitted forms by report type, date, and keywords.
- date - uses the default date class from JavaScript.
- delayedUploadController - Contains ability to get online status, choose form to sync, get new data, and upload new data.
- delayedUploadView - Displays the sync graphic when syncing.
- syncController - Contains ability to check if synced, send updated models, and receive updated models.
- syncFormModel - Receives the unique identification number.

2.2.2 Function Descriptions

All classes will have getter and setter methods to give access to the many variables. These functions will be public so information about the reports can be accessed anywhere. Examples include:

```
-function isValid: checks the report to ensure that it is valid to be saved. If it is not, highlights
all invalid inputs.

-function submit: allocates memory for the report and saves it to the local database.

-function filterRecordsWithQuery(Query query): Given a query, filter the records to show all records
that conform to the query.

-function displayRecordWithID(int id): Given a report ID, display the report associated with that ID.

-function getRecordsForBrowse(int recordType, String town): Returns all reports of the given type
for the given town.

-function getOnlineStatus(): Checks to see if device is connected to the internet.

-function addFormToSync(UID): Given the UID, places form in queue to be uploaded
```

`-function getNewData(): Checks form to get new data.`

`-function uploadNewData(): Uploads any new data to database.`

`-function showSyncGraphic(): Displays the syncing graphic as upload is in progress.`

`-function synced: returns boolean of is synced or not synced.`

`-function isSynced(): Checks to see if form is synced with the database.`

`-function sendUpdatedModels(): Sends the updated models to the database.`

`-function getUpdatedModels(): Requests the updated models from the database.`

3 SYSTEM BEHAVIOR

3.1 FORM ENTRY

3.1.1 Principle Action: Enter Data on Form

The user enters the field data for the objects. If the user enters inappropriate data, system prompts the user with an error message and request the correct data format to be entered. Once the required field values are entered for the object it is ready to be uploaded (refer to section 3.3).

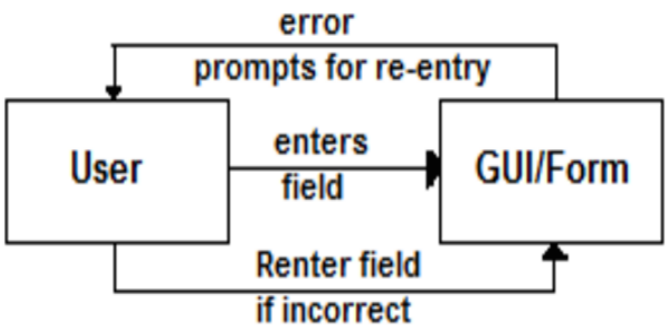


Figure 3: Sequence diagram for entering data into form.

3.2 BROWSING AND SEARCH

3.2.1 Principle Action: Browse Records

To browse records, the main controller accesses the database for all stored records. The database returns the records, and they are stored in Report objects which are displayed as icons for the user.

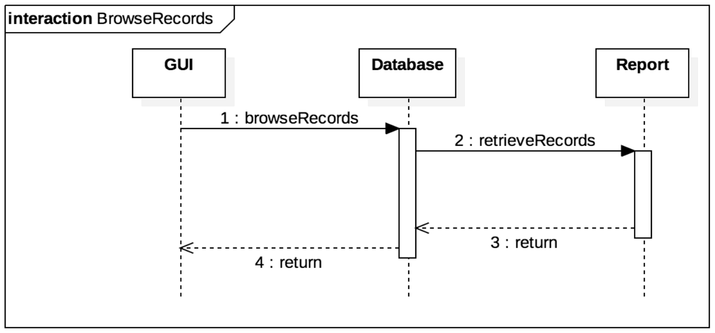


Figure 4: Sequence diagram for retrieving reports for browsing.

3.2.2 Principle Action: Search for Reports

To search for a report, a query object is created from a search string inputted by the user. Through an Ionic plugin, the query filters out the previously loaded reports that are not relevant according to the query string.

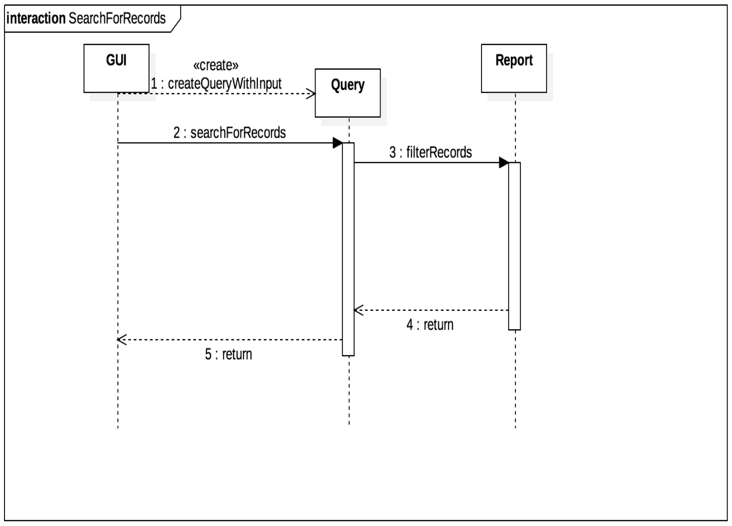


Figure 5: Sequence diagram for filtering stored reports.

3.3 DELAYED UPLOAD

3.3.1 Principle Action: Add Document to be Uploaded

To add a document to be uploaded, control is passed from the main controller to the DelayedUploadController where .addFormToSync is called, thus passing control to the SyncController where the document is added to a queue.

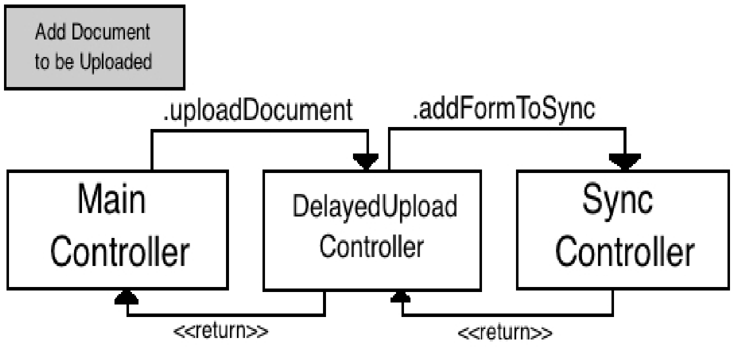


Figure 6: Sequence diagram for adding a document to be uploaded.

3.3.2 Principle Action: Upload Document

To upload a document, control starts with the SyncController and is passed to the SyncFormModel. The document is then uploaded to the database and the SyncController informs the DelayedUploadController of its success. The DelayedUploadController then updates the status icon for that document using DelayedUploadView.

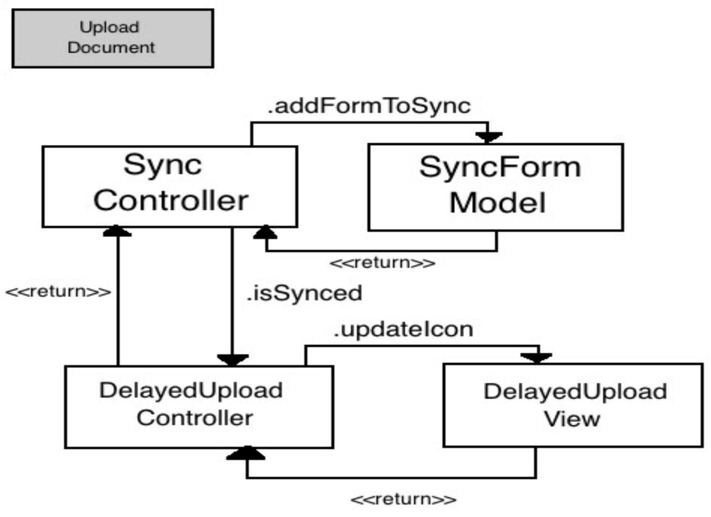


Figure 7: Sequence diagram for uploading a document.

4 DATA DESIGN

4.1 DATA DESCRIPTION

Data and their fields (Restaurant, RestaurantInspection, ReportType, Well, QualityReport, Violation, Location, Septic, PumpingReport) are collected from users and stored in database. Data is searched using Query object and its fields. Search results are generated from the database and displayed to the user.

4.2 DATA DICTIONARY

Data Object	**Field**	**Type**
Query		
	keyword	String
	minDate	Date
	maxDate	Date
	restaurant	boolean
	well	boolean
	septic	boolean
User		
	name	String
	username	String
Restaurant		
	restaurantName	String
	phoneNum	String
	permitNum	int
	personInCharge	String
	operationType	String
	inspector	User
RestaurantInspection		
	repeat	boolean
	violations	List<Violations>
	timeIn	Time
	timeOut	Time
	HACCP	boolean
	riskLevel	String
	discussionPC	String
	correctiveActReq	boolean
	correctiveActions	List<String>

	numViolations	int
	inspectionType	String
ReportType		
	date	Date
	location	Location
	owner	String
Well		
	wellDepth	double
	wellDriller	Name
	reports	List<QualityReport>
QualityReport		
	yield	double
	quality	Quality
Violation		
	description	String
	code	int
	violationName	String
Septic		
	ownerLocation	Location
	ownerPhone	String
	cesspools	boolean
	septicTank	boolean
	tightTank	boolean
	greaseTrap	boolean
	otherDescription	String
	reports	List<PumpingReport>
PumpingReport		
	gallonsPumped	double
	teeFilterPresent	boolean
	teeFilterCleaned	boolean
	condition	String
	PumperName	String
	vehicleLicenseNum	int
	pumperCompany	String
	disposalLocation	Location
	haulerSignature	String
	haulerDate	Date
	receiverSignature	boolean
	receiverDate	Date

Table 1: Data dictionary

5 USER INTERFACE DESIGN

5.1 DESCRIPTION OF THE USER INTERFACE

The user interface will consist of various menus through which the user can interact with the tablet application. These menus will include a ”Browsing and Search” menu, a ”Form Entry” menu, and a ”Delayed Upload” status bar. Each menu will contain fields for navigating and performing a specified task. The user will interact with the menus through the device’s touch screen.

Each menu will consist of various GUI components, such as buttons, labels, text fields, and scroll wheels. These components will be arranged in such a way that the user will be able to quickly navigate through each menu, complete forms, search and browse through the database. The top of the tablet screen will have a header including an ”add new” button, a search icon, and a drop down menu. The drop down menu will show options for logging out or sorting the thumbnails on the screen.

A detailed description of these menus and their interactions with one another will be described in the following section.

5.2 SCREEN OBJECTS & ACTIONS

5.2.1 Browsing and Search

Upon entering the browsing and search menu, a search text or icon will always be available at the header. When tapped, the search bar will expand and a touch keyboard will appear. The search results will display the search terms and the surrounding text will be displayed along with the search terms. The search terms will be in highlighted in bold font.

This design for the search results is influenced by Google's search results pages. Any information that appears next to the search terms may be relevant information that may be accessed without having to open up the full inspection form.

The search will also provide suggestions as the user types out the term. There will be an option to sort by category specifically or a date. When the user selects a form that they have found, options to view, edit, email, and print will appear.

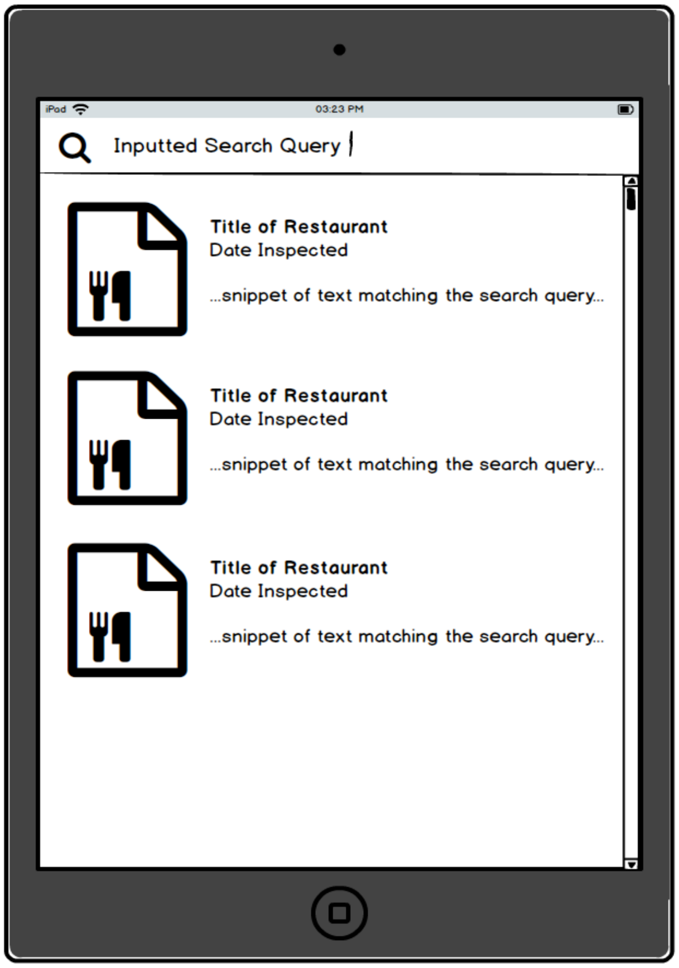

iPod
03:23 PM
Inputted Search Query
Title of Restaurant
Date Inspected
...snippet of text matching the search query...
Title of Restaurant
Date Inspected
...snippet of text matching the search query...
Title of Restaurant
Date Inspected
...snippet of text matching the search query...

Figure 8: Search screen

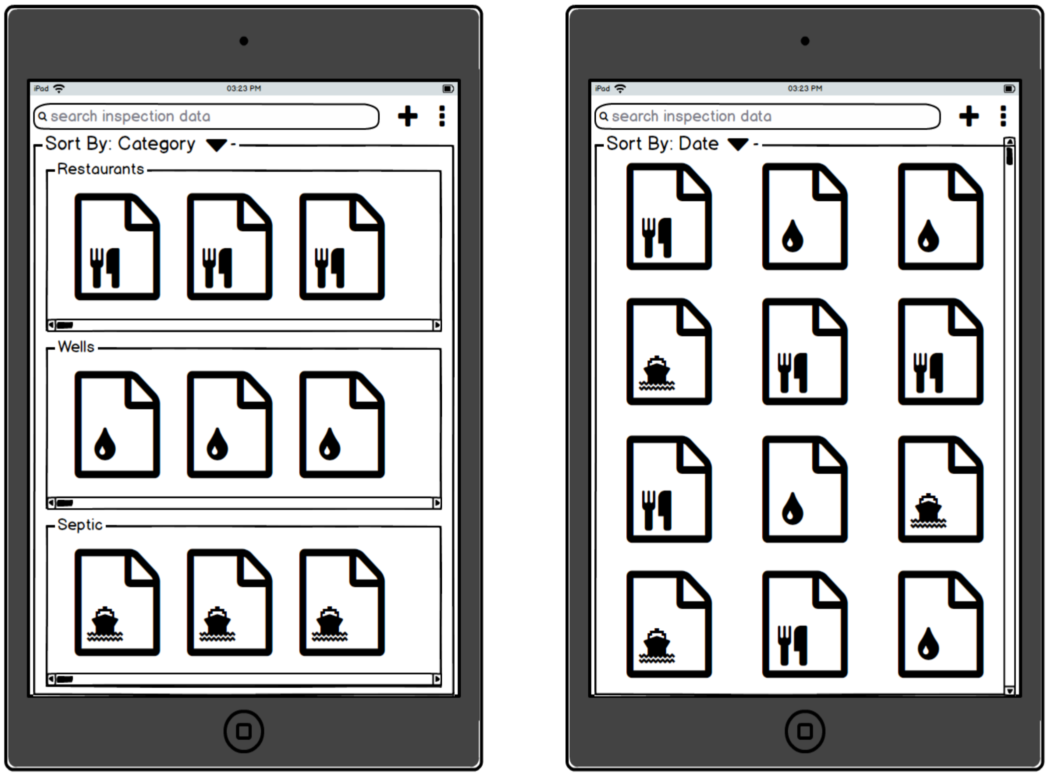


Figure 9: Browse by category/date screens

Three categories will be arranged underneath the header. These categories are restaurants, wells, and septic tanks. A row of documents will present each of these categories and each one will have its own scroll bar. Each document is represented by a thumbnail and will be sorted by most recent starting from left to right.

This design for browsing is inspired by Netflix's browsing menus on both their website and mobile applications. The user is greeted with the most recent information entered into the tablet database and shows documents from all three categories of the project. Thumbnail icons are a great way to show the type of documents and any vital information about that document.

When the user taps on one of the three rows of categories, it expands the tapped category to the full screen and displays all of the document thumbnails in that category. The document thumbnails are sorted by most recent starting left to right while going top to bottom. The user can scroll vertically to view more document thumbnails. A user can tap on a thumbnail on the screen and options to view, edit, email, and print will be displayed.

This option for browsing may be a quicker way to find documents since it filters by document type to start and displays more thumbnails on the screen at once compared to the rows that are displayed by default.

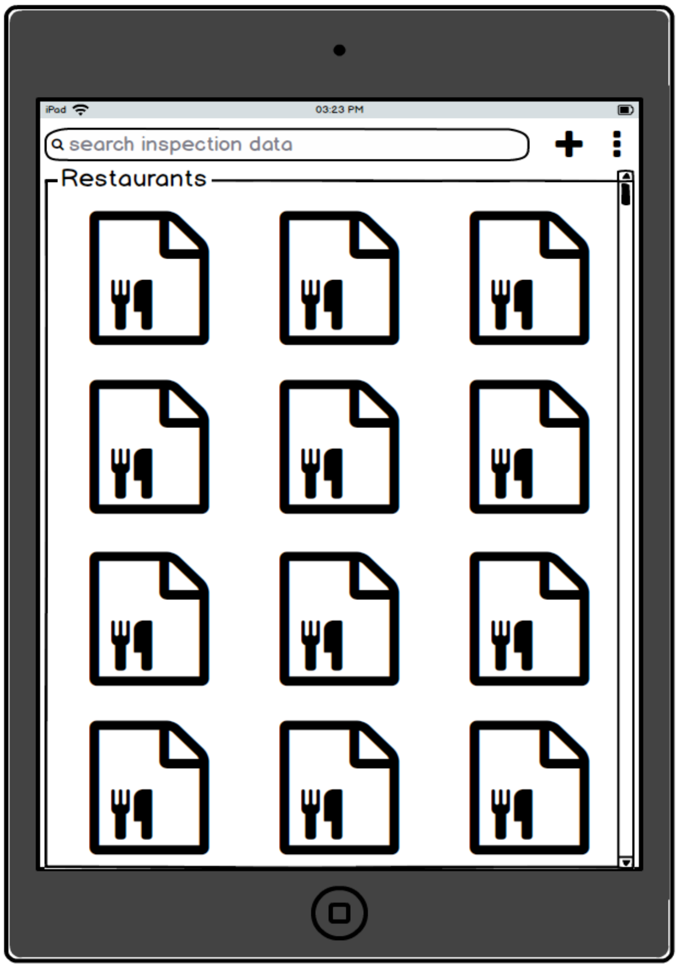


Figure 10: Specific category chosen screen

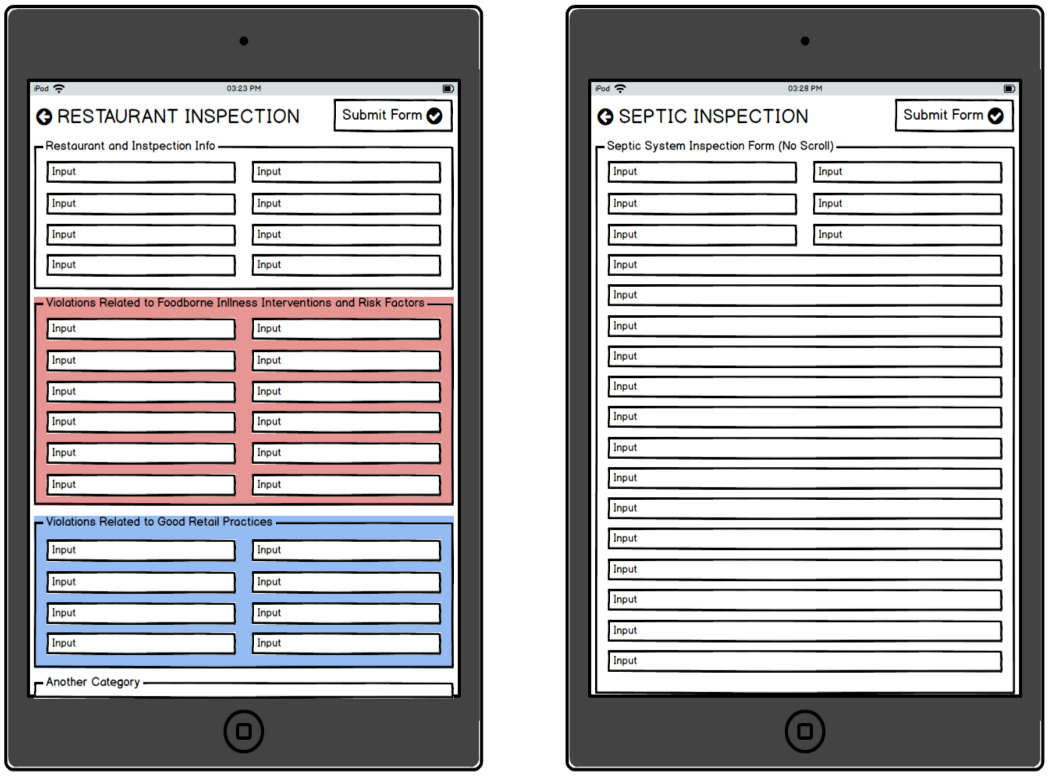

iPad 03:23 PM
RESTAURANT INSPECTION
Submit Form
Restaurant and Instpection Info
Violations Related to Foodborne Inllness Interventions and Risk Factors
Violations Related to Good Retail Practices
Another Category
Input
iPad 03:28 PM
SEPTIC INSPECTION
Submit Form
Septic System Inspection Form (No Scroll)
Input

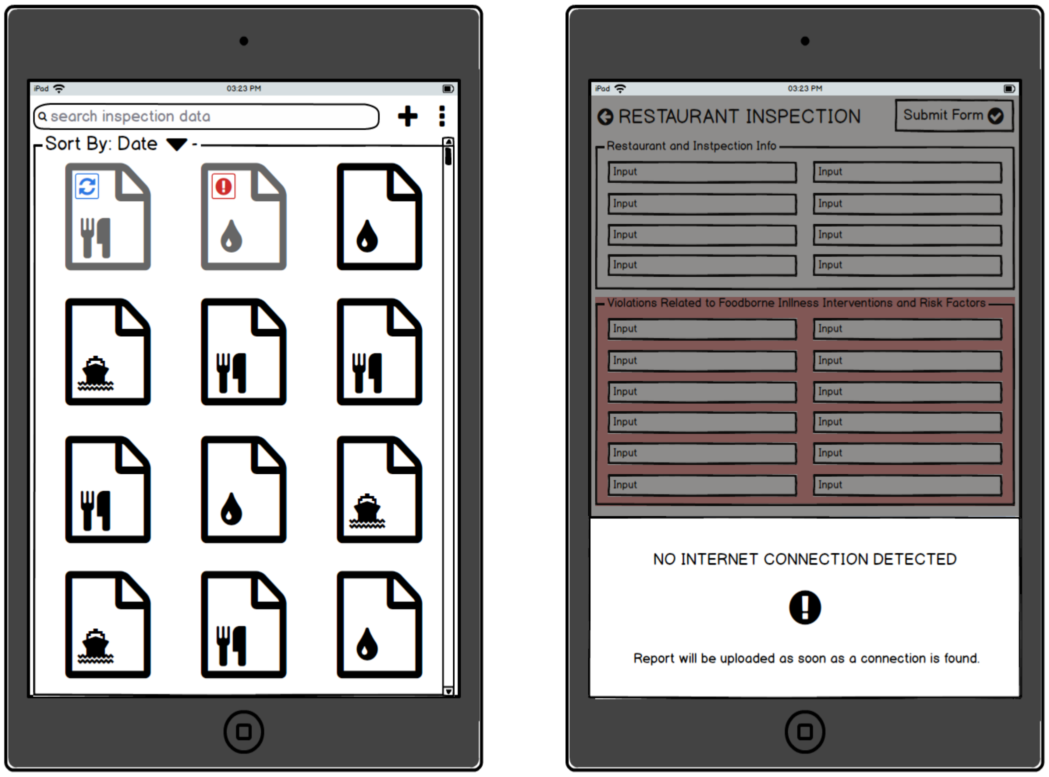
Figure 11: Restaurant Inspection/Septic Report screens

5.2.2 Form Entry

Upon choosing a form, the tablet displays a simple blank form with fillable text fields (see Figure 6). For the restaurant inspection form, information is split into different sections. Each section will be color coded similarly to the paper form currently used.

When a field is tapped, the tablet's on-screen touch keyboard will appear. The whole fillable form is displayed on one screen and the user will scroll vertically.

A ”complete” button will be shown on the tablet screen. When the button is tapped, the data entered will be applied to a PDF form and several buttons including print, edit, email, and exit will appear on the screen. A sync status will also appear here.


iPod
03:23 PM
search inspection data
Sort By: Date
RESTAURANT INSPECTION
Submit Form
Restaurant and Instpection Info
Input
Violations Related to Foodborne Inllness Interventions and Risk Factors

NO INTERNET CONNECTION DETECTED

Report will be uploaded as soon as a connection is found.

(a) Sync icons (b) Submitting form with no internet access

Figure 12: Delayed Upload Screens

5.2.3 Delayed Upload

There will be 2 different icons shown on each document thumbnail. Each of these icons will have a distinctive shape and color. A blue sync icon means an upload is in progress. A red sync icon means upload has failed and will try again when network is available. No icon means the file is successfully on the server.

The auto-hiding status bar at the bottom of the screen appears when syncing (blue) or when syncing has failed (red) and will be hidden when there is nothing to be synced.