SOFTWARE REQUIREMENTS SPECIFICATION (SRS) FOR

320 Green Team Project

Version 2.0
November 9, 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

1.1 PURPOSE

Replace the current physical method of archiving Board of Health data from restaurant inspections, well-water quality reports, and septic tank pumping reports with a new digital method. This document provides a high-level overview of the entire system before going into detail on the tablet application portion's form entry, browsing and search, and delayed upload functionality.

1.2 SCOPE

This project includes multiple sub-projects and systems.

This new digital method requires a database to hold different types of inspection results and reports, including data on restaurants, well-water quality, and septic tanks. The database must be searchable and sortable by keywords or established parameters, thus improving accessibility and simplifying data analysis. Storing this data digitally will allow the customer to better analyze patterns and create a more accessible and interactive location for the data to reside. It will act as an intuitive tool to help the inspector use and access data in the future. To access this database, a user interface must be created to provide fast access and retrieval of information. This application focus on usability, with a sub-portion designed for portability. This application is meant to be used specifically in a municipality's Board of Health office environment. The application will include a convenient and easy method to search, sort, and analyze data from the restaurant inspections, well-water quality reports, and septic tank pumping reports. Septic tank and well-water quality data should also be viewable on a geographical map with the ability to show changes over time. This will illuminate differences and trends in the data that may help explain and predict them. The cataloging of restaurant health inspection data could be facilitated by a form on a portable application (i.e. mobile, tablet). This portable application should be designed to locally store the entered data and convert it to a form readable by the database so it can add the record to the rest of the database when connected.

Overall, creating and structuring a database for storing various health data, and providing easy-to-use tools for accessing it, will assist the customer in his inspections and simplify the retrieval of key information he may find interest in at a later date.

1.2.1 Sub Projects

- Central Database
- Form Entry Application for Tablet
- Desktop Application

Central Database:

A central database with constant access to data will ensure the integrity of the entire system. The three distinct data entities identified were: Well-water quality reports, septic system pumping reports, and restaurant inspections, which will all need to be differentiated by town. The attributes of the models will correspond to the required fields that the customer must enter during inspections. The data for the restaurant inspections shall contain all fields that can be found on the restaurant inspection form. The data for well reports must contain all of the relevant fields, such as water quality, measurements, address of water well, flow rate, name of the company that drilled the well, and the permit and lot number. Similarly, the data for the septic pumping report will hold information such as the address, lot number, size, capacity, condition of the tank, date of last inspection, and quality of gallons pumped in and out. The type of data

stored must be able to be changed in the event of new legislation.

The first of two sub-projects for creating this database are to create a basic database that provides basic queries, such as: creating, reading, editing, and deleting records safely and securely. The second sub-project is to create and design a method to federate the database so that other portions are able to access the database structure.

Form Entry Application for Tablet:

This application will be focused on usability in order to make it as simple as possible to enter the data. The application will be able to store data locally on the device; and the ability to upload to the office-based application noted (e.g. desktop application or web-application). This app will replace the forms that the customer requires for every inspection by including fields for all relevant information. For security purposes, the app will have a log-in screen for user authentication. This application will be able to either send the document to a printer or email it on the spot, as well as be able to receive an e-signature. The application should allow the user to view relevant information based on data that is associated with the user's current task that has been submitted previously. For example, when the user is going to inspect a restaurant, they should be shown previous inspection results from that restaurant.

The tablet application will have the following main sections that correspond with the functionality above:

- Data Entry with a user interface that allows for dynamic flow through the inspection process.
- Geographic Information System (GIS) to locate well, septic, and restaurant locations.
- Browsing and search of data in order to review previously entered reports for analysis purposes.
- User authentication to verify identity of the user.
- Form printing in order to leave any necessary paperwork after an inspection.
- Delayed upload to save and move data without needing constant internet access

Application (Usable, Connected to Database):

An application will be created for viewing all past records stored in the database. The application should include three different sections for restaurant, wells, and septic systems. The application will need to communicate with the central database containing all information of past records. The application will provide an interface for querying data, including searching by location or violation. The application will contain user authentication that, once validated as an administrator, will allow for the submission of new records. Without log-in credentials, the information will be visible, but immutable. The application will use the accepted regulation forms as a base to record data from new restaurant inspection, water, or septic system reports and then send this new data to the relevant database. Additional functionality of this application includes:

- Reporting and search that includes many enter-able parameters to view and categorize reports
- User authentication so that inspectors can add report data and constituents may view public records.
- Data mapping to view data in relation to maps of corresponding towns.
- Printing of reports or search results for when physical copies are desired.

1.3 INTENDED AUDIENCE

This system is intended for health inspectors who want to use a digital alternative to record and store information as opposed to paper storage. Specifically, the system is being made for the health inspector of the towns of Leverett and Sunderland, Massachusetts.

1.4 DEFINITIONS, ACRONYMS, AND ABBREVIATIONS

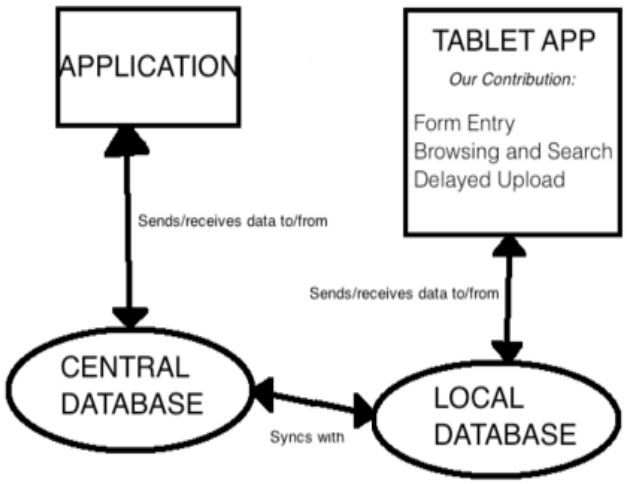
The product's name is Health.e standing for Health Electronics, as it is designed to digitize information on health inspections, well water and septic tanks.

Term or Acronym	Definition
DFD	Data Flow Diagram
SRS	Software Requirements Specification
GIS	Geographic Information System

2 OVERALL DESCRIPTION

2.1 PRODUCT PERSPECTIVE

This product will be replacing the primary user's current pen and paper system for documenting inspections. Rather than filling out paper forms on site and handing the inspectee a carbon copy of the form, the tablet app will allow the inspector to collect the same data and print a standard form from a portable printer on site for the inspectee to review. Also, rather than storing paper files in a filing cabinet that may never be looked at again, the health inspector may, with our product, review previous forms with ease by searching for them with our comprehensive browsing and search functionality. This way, records may be found in a matter of seconds. Another aspect of this product will be the delayed upload in which forms that have been filled out on the tablet and stored locally will be synced with the central database so that they may be integrated with the main application. This eliminates the need for the health inspector to "file" each form after completing it.
The form entry, browsing and search, and delayed upload functionalities are all subprojects for the entire tablet app, which is, in and of itself, a subproject of the main project. Our contribution to the main project is depicted below:



2.2 PRODUCT FUNCTIONS

2.2.1 Data Entry (Forms)

1. The user chooses one of the following inspection forms to fill in: restaurant, septic tank, or water well.
2. The tablet then displays the chosen form to the user with blank, fillable forms.

2.2.2 Browsing & Search

1. The user types in a location of the desired restaurant, septic tank, or water well.
2. The tablet displays records from its database relevant to the search requirements.

2.2.3 Delayed Upload

1. The tablet synchronizes with the external database and uploads either specific or all records from the tablet to the external database once a connection has been established.

2.3 USER CHARACTERISTICS

The form entry, browsing and search, and delayed upload parts of the tablet will primarily be used by health inspectors. They will have full access to all functional aspects listed in section 2.2. Unregistered guests will have limited, read-only access to the main application of the project while only the registered health inspectors may access the tablet application.

2.4 OPERATING ENVIROMENT

The form entry, browsing and search, and delayed upload aspects of this project will all exist on the same platform: a tablet. The tablet should have a small amount of memory allocated to the app so that data records may be stored locally until uploaded to the central database.

2.5 CONSTRAINTS

2.5.1 Data Entry (Forms)

The form entry itself is primarily based on inspection forms created by the state and used by the town. There must be an entry field for each piece of data corresponding to the required sections of the current inspection forms. Also, there will be a finite number of form entries the user can generate on the tablet since the tablet will have a limited amount of memory space. With recent tablet device improvements, the chances of filling up the memory on the tablet is low. With a good design of the entire project, modern day memory capacities will suffice to hold the data required.

2.5.2 Browsing & Search

Browsing and searching features are constrained by the design of the database itself. With a good design and structure, these features will be user-intuitive and minimal effort will be needed by the user to use these features. Browsing and searching the form entries is constrained by the number of form entries in the database on the tablet.

2.5.3 Delayed Upload

The delayed upload is constrained by the health inspector's access to WiFi or some form of manual docking technology. Either method may be used in order for the local memory on the tablet to sync with the central database, but one must exist.

2.6 USER DOCUMENTATION

There will be a GitHub Wiki that will help guide the user through all of the basic functionality of the tablet app, and in general, the application as a whole. The Wiki will be user friendly, easily comprehensible, and will touch on all of the important features of the app that the primary user should be familiar with.

2.7 ASSUMPTIONS & DEPENDENCIES

2.7.1 Data Entry (Forms)

Our implementation of the form entry on the tablet app depends on how the central database is constructed. Each field that the database contains for each type of form must be represented in the UI of the tablet app so that the health inspector may enter all of the fields correctly. The assumption here being that any changes made to the physical forms will be updated accordingly in the database, and thus updated in the digital forms as well.

2.7.2 Browsing & Search

Our implementation of browsing and search on the tablet app relies on an internal database that holds previous records. We assume that the internal database is synced with the main database, so that the files being searched for can be found within the tablet. There is a possibility that files that have been removed from the tablet may be searched for, but such a case will be handled by a file not found message.

2.7.3 Delayed Upload

We are assuming that the health inspector will have WiFi access in order for the tablet app to upload data from its local database to the central database via a syncing process. In the case that this assumption is incorrect and the user does not have WiFi access, a manual docking process will have to occur between the tablet and the health inspector's computer on which the main application exists. This assumes that the health inspector has the proper cable to connect the tablet to said computer, but this is a reasonable assumption since such a cable typically is included with the purchase of a tablet since its serves the dual purpose of a charger. The delayed upload also depends on the construction of the database and its fields as well as the application itself since it must sync with said application.

3 EXTERNAL INTERFACE REQUIREMENTS

3.1 USER INTERFACES

The user interface for the system will be an app based application on a tablet. The user will enter log-in parameters as user authentication. The user will then be able to select which records to access. The user will be able to enter record data in the app on the replicated forms for the inspection. The user will also be able to send the document to a printer or email it. The app will also have the ability for the user to browse and search for information on the app. The user will be able to select for a delayed upload as well.

3.2 HARDWARE INTERFACES

The hardware interfaces of the system will be run on an android device. An internet connection will allow the software interfaces to connect to the internet and sync with records in the database. The tablet will also contain a local database allowing for complete offline access of previous record entry.

3.3 SOFTWARE INTERFACES

3.3.1 Operating System

The software is being designed to run on Ionic. Ionic is a tool that is written in html, css, and javascript allowing for a web-app, ios-app- and android-app to be developed at once.

3.3.2 Web Server

The software will be able to sync with a database as well as contain its own personal database. This will allow for consistent access to previously recorded data. The web server and local database will be synchronized with a consistent data-model.

3.3.3 Database

The software will use a federated database to store all the data. This federated database will allow for the decentralization of the two towns, Sunderland and Leverett but also provide access to all records through a common API.

3.4 COMMUNICATION INTERFACES

3.4.1 Form Interface

The application will provide electronic entries that the user needs. These entries will be modeled after the forms provided by the customer and provide the ability for the customer to submit an entry in any order required.

3.4.2 Web Communication

The application will be able to wirelessly sync with the database. This feature will not be the only syncing mechanism but will provide an alternative to how data is uploaded. All syncing will interact through the web application which will allow the database to only interact with a single entity (the web application) thus preventing any write conflicts between the tablet and web application.

3.4.3 USB Communication

The application will be able to sync through a usb cable with the database. This hard wire connection will provide full autonomy for the customer to upload specific forms to the web application. The USB connection will also be able to read from the computer in order to update attributes on the tablets local database.

4 NON-FUNCTIONAL REQUIREMENTS

4.1 SOFTWARE QUALITY ATTRIBUTES

The most important aspect of the final product is that it must be extremely user-friendly. Operating the software should be a simple and intuitive process. A new user should be able to learn how to operate the software very quickly. Learning how to use this software is not the job, but the tool to accomplishing the job easily and effectively.

4.2 PERFORMANCE REQUIREMENTS

The software must operate efficiently enough such that it does not cause delay during an inspection. Operation of the software should be quick and responsive in order to satisfy the user's needs.

4.3 SAFETY REQUIREMENTS

We must operate under the assumption that the application may be offline for some amount of time. During this time, all data saved on the tablet must remain organized and secure. During any instance of Delayed Upload, data must remain intact and the connection must be secure such that no corruptions or issues arise.

4.4 SECURITY REQUIREMENTS

It is vital that data entry is only accessible to authorized users. User authentication should ensure that only administrators can enter information into the database. While the data on the tablet will eventually become open to the public, this part of the application should be restricted to inspectors.

5 FUNCTIONAL REQUIREMENTS

5.1 DATA ENTRY (FORMS)

5.1.1 *Use Case 1*

Title: **Fill out septic tank inspection form**
Primary Actor: User
Secondary Actor: System
Trigger: User selects septic inspection form.
Precondition: User successfully logged in and accessed the form entry user interface for the septic tank.
Postcondition: Data is stored to be submitted to a central database

Main Scenario:

1. System presents modifiable septic tank information form.
2. User fills out each required field of the form.
3. User submits the form on tablet.
4. System determines that each required and non-required field entered is valid.
5. System creates an accessible local copy.
6. System displays a response when successfully entering form.
7. User safely closes out of process (data is preserved).

Exception Scenario:

1. User forgets to enter information in 'required field'.
   (a) System notifies user of empty fields and does not submit.
2. User enters an invalid form entry.
   (a) System displays invalid form entry and does not submit.
3. System shuts down during data submission.
   (a) System can show user whether or not data was submitted successfully.

5.1.2 *Use Case 2*

Title: **Fill out well water inspection form**
Primary Actor: User
Secondary Actor: System
Trigger: User selects well water inspection form.
Precondition: Successfully logged in and accessed the form entry user interface for water well inspection.
Postcondition: Data is stored to be submitted to a central database

Main Scenario:

1. System presents modifiable well water information form.
2. User fills out each *required* field of the form.

3. User submits the form on tablet.
4. System determines that each required and non-required field entered is valid.
5. System allocates memory and creates an accessible local copy.
6. System displays a response when successfully entering form.
7. User safely closes out of process (data is preserved).

Exception Scenarios:

1. User forgets to enter information in 'required field'.
   (a) System notifies user of empty fields and does not submit.
   (b) User presented with partially filled out form.
2. User enters an invalid form entry.
   (a) System displays invalid form entry and does not submit.
   (b) User presented with partially filled out form.
3. System shuts down during data submission.
   (a) System shows user whether or not data was submitted successfully.

5.1.3 *Use Case 3*

Title: **Fill out restaurant inspection form**
Primary Actor: User
Secondary Actor: System
Trigger: User selects restaurant inspection form to fill out.
Precondition: User has passed login authentication, and has chosen to submit data for a restaurant inspection.
Postcondition: Data is stored to be submitted to a central database

Main Scenario:

1. The page for restaurant inspection form is shown.
2. User records location and name of restaurant
3. User fills out fields in form to record data form inspection
4. User clicks save button.
5. Data is stored locally

Exception Scenario:

1. User forgets to enter information in 'required field'.
   (a) System notifies user of empty fields and does not submit.
   (b) User presented with partially filled out form.
2. User enters an invalid form entry.

(a) System displays invalid form entry and does not submit.

(b) User presented with partially filled out form.

3. System shuts down during data submission.

   (a) System shows user whether or not data was submitted successfully.

5.2 BROWSING & SEARCH

5.2.1 *Use Case 1*

Title: **Search for prior inspection report**
Primary Actor: User
Secondary Actor: System
Trigger: User wants to search for a list of inspection reports matching an entered keyword.
Preconditions: User has logged in and is on the inital lookup report screen which contains a list of all inspection reports.
Postcondition: System displays list of prior reports matching search term.

Main Scenario:

1. User enters a keyword to search
2. System queries external database with location and retrieves records that contain the keyword
3. System displays a list of retrieved records and their dates, sorted by date

Exception Scenarios:

1. System cannot connect to database

   (a) Screen displays connection error message and asks user to try again later

   (b) User selects 'Ok' button and system returns to the lookup report screen

2. None of the records match the search keyword

   (a) Display message, "No reports match the specified keyword"

   (b) User clicks 'Ok' and returns to the initial lookup report screen

5.2.2 *Use Case 2*

Title: **Narrow down list of inspection reports**
Primary Actor: User
Secondary Actor: System
Trigger: User wants to select criteria to narrow down a list of all inspection reports.
Preconditions: User has logged in and is on the initial lookup report screen which contains a list of all inspection reports.
Postcondition: System displays list of prior reports matching the selected criteria.
Main Scenario:

1. User selects criteria to narrow down all the inspection reports
2. System queries external database with location and retrieves records that match the criteria

3. System displays a list of retrieved records and their dates, sorted by date

Exception Scenarios:

1. System cannot connect to database
   (a) Screen displays connection error message and asks user to try again later
   (b) User selects 'Ok' button and system returns to the lookup report screen
2. None of the records match the search keyword
   (a) Display message, "No reports matching that criteria"
   (b) User clicks 'Ok' and returns to the initial lookup report screen

5.2.3 *Use Case 3*

Title: **View an inspection report**
Primary Actor: User
Secondary Actor: System
Trigger: User wants to view a specific inspection report.
Preconditions: User has logged in and is on the lookup report screen which contains a list inspection reports.
Postcondition: System displays the selected inspection report.
Main Scenario:

1. User selects an inspection report
2. System displays all data contained in the selected inspection report

Exception Scenario:

System cannot connect to database

1. Screen displays connection error message and asks user to try again later
2. User selects 'Ok' button and system returns to the lookup report screen

5.3 DELAYED UPLOAD

5.3.1 *Use Case 1*

Title: **User attempts to upload a specific piece of form data.**
Primary Actor: System
Secondary Actor: User
Trigger: User selects septic inspection form to fill out.
*Precondition:*User has completed and saved some type of form data, and is associated with some online database. User is on the start page and has signed in successfully. Alternatively, User opts to send form data immediately after saving to device and begins at step 2.
*Postcondition:*Form data has not already been submitted to the database and is added successfully. If conditions are not correct for proper form data upload, data is not uploaded.

Main Scenario:

1. User navigates the form data on the application to find the data they desire to upload.

2. User indicates that they want to upload the form data.
3. System verifies that the form data is acceptable for submission to online database.
4. System verifies that internet connection is available.
5. System verifies that connection to associated database is available.
6. System verifies that given form data has not already been submitted to the database.
7. System asks the user for confirmation to send the form data.
8. User gives confirmation of intent to send the form data
9. System adds the new form data to the database using the internet connection available.
10. System notifies the user of successful upload of the data.

Exception Scenario:

1. Form data is not acceptable for upload.
   (a) System notifies the user that the form data is not acceptable for upload. System specifies which fields in the form data are unacceptable and asks that they be fixed before submission.
   (b) User is returned to the form's viewing page.
2. No internet connection is available.
   (a) System notifies the user that internet connection is not available and to try again once they are within access range.
   (b) User is returned to the form's viewing page.
3. Only internet connections that are available are not Wi-Fi connections.
   (a) System asks the user to verify that they would like to send the connection on the mobile network.
   (b) If user accepts, use case proceeds as normal at step 4.
   (c) If user refuses, system notifies the user to try uploading again once they are within access range.
   (d) User is returned to the form's viewing page.
4. System is unable to connect with the database.
   (a) System notifies the user that it is impossible to connect to the database and to verify that the database is online.
   (b) User is returned to the form's viewing page.
5. Form data has already been submitted to the database.
   (a) System notifies the user that the form has already been uploaded before.
   (b) User is returned to the form's viewing page.
6. User declines the confirmation to send the data.
   (a) Form data is not sent. User is returned to the form's viewing page.
7. No internet connection is available.
   (a) System notifies the user that internet connection is not available and to try again once they are within access range.
   (b) User is returned to the form's viewing page

5.3.2 *Use Case 2*

Title: **Batch upload all forms that are not already uploaded.**

Primary Actor: System

Secondary Actor: User, Databse

Trigger: User selects septic inspection form to fill out.

Precondition: User has completed and saved some type of form data, and the device is associated with some online database. User is on the start page and has signed in successfully. Alternatively, setting for automatic upload is on and System engages automatically, starting at step 2.

Postcondition: Form data that has not already been submitted to the database and is acceptable for upload is added successfully. If conditions are not correct for proper form data upload, data is not uploaded.

Main Scenario:

1. User indicates that they would like to perform a batch upload of form data.
2. System verifies that internet connection is available.
3. System verifies that connection to associated database is available.
4. System cross-references form data on the device and on the database to find which data is on the device and not in the database.
5. System displays a list of the eligible form data on the device and requests the user confirms the data.
6. User selects the data they wish to upload to the database and confirms to send those forms.
7. System checks each data that is desired to be uploaded is acceptable to be added to the database.
8. System sends acceptable form data to the database to be added.
9. System notifies the user of which form data was successfully sent to the database.

Exception Scenario:

1. No internet connection is available.
   (a) System notifies the user that internet connection is not available and to try again once they are within access range.
   (b) User is returned to the main viewing page.
2. Only internet connections that are available are not Wi-Fi connections.
   (a) System asks the user to verify that they would like to send the connection on the mobile network.
   (b) If user accepts, use case proceeds as normal at step 3.
   (c) If user refuses, system notifies the user to try uploading again once they are within access range.
   (d) User is returned to the main viewing page.
3. System is unable to connect with the database.
   (a) System notifies the user that it is impossible to connect to the database and to verify that the database is online.
   (b) User is returned to the main viewing page.
4. All form data on the device is also on the database.

(a) System notifies the user that there is no data to be upload to the database.

(b) User is returned to the main viewing page.

5. User declines to upload data.

   (a) User is returned to the main page.

6. Some form data is not acceptable to be uploaded to the database

   (a) Any data that is not acceptable is removed from the list of data to be uploaded. After batch upload is finished, a notification is sent with a list of any data unacceptable to be uploaded.

7. No internet connection is available.

   (a) System notifies the user that internet connection is not available and to try again once they are within access range.

   (b) User is returned to the main viewing page.

8. System is unable to connect with the database.

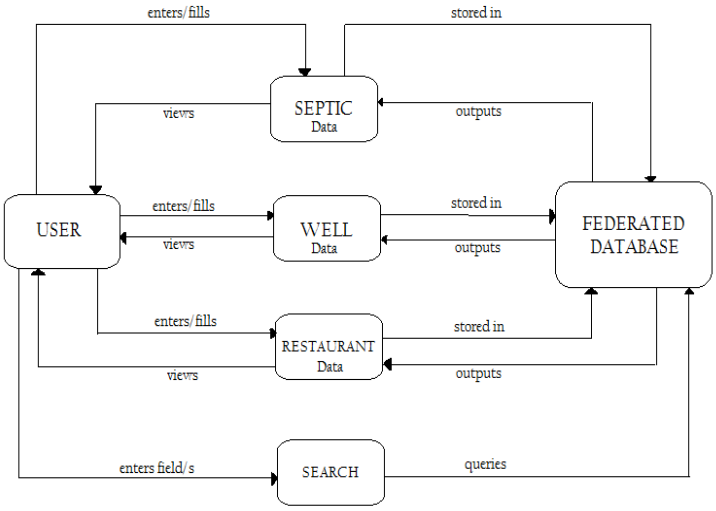
   (a) System notifies the user that it is impossible to connect to the database and to verify that the database is online.

   (b) User is returned to the main viewing page.

6 DATA MODELING

6.1 DATA DESCRIPTION

Data from three six objects and their respective fields are collected from the user and stored in the database. Objects are searched using fields. Search results in data in the form of objects being output from the database.



6.2 DATA OBJECTS

Three different data objects and their fields are stored in the database; septic inspection form, water well inspection form, and restaurant inspection forms. Users use search object and its field to query the database.

- Septic tank: Septic tank objects holds fields and data from septic tank inspection forms. These objects are stored in the database and can retrieved using search options.
- Water Well: Water well object holds fields and data from water well inspection forms. These objects are stored in the database and can retrieved using search options.
- Restaurant: Restaurant object holds fields and data from restaurant inspection forms. These objects are stored in the database and can retrieved using search options.
- Search: Search object holds the fields entered by the user during search process. Database matches the fields from the search object to the corresponding stored objects and fields to output the results.

7 TEST PLAN

7.1 OBJECTIVES

The objective of the test plan is to ensure a high level of confidence in the correctness and usefulness of the project deliverables.

7.2 TESTING STRATEGY

The strategy for testing the softwares data entry, browsing and search, and delayed update features is a combination of automation testing and manual testing. Functional requirements of the listed features are to be tested by using automation testing. Manual testing will only be used for features that may not be testable by automation. Certain components of these features may involve more intensive interaction between the tablet and the user and these components will need validation by testers and developers.

7.3 SCOPE

Testing for this section will be conducted throughout the development life cycle of the product.

7.4 FEATURES TO BE TESTED

Test cases will cover all the features of the software covered in the SRS. It will include the exception cases in the use cases so as to ensure that the product behaves in the anticipated manner.

TEST ID	**SRS Document Reference**	**Features To Be Tested (Test Description)**
001	5.1.1	Validate the data entry form feature for septic tanks. Test cases will ensure that the user is able to input and upload the form in the expected manner. Test cases will also test for exceptions: 1) User misses a required entry, 2) User inputs invalid entry 3) System shutdown during data upload.
002	5.1.2	Validate the data entry form feature for water well. Test cases will ensure that the user is able to input and upload the form in the expected manner. Test cases will also test for exceptions: 1) User misses a required entry, 2) User inputs invalid entry 3) System shutdown during data upload.
003	5.1.3	Validate the data entry form feature for restaurant inspections. Test cases will ensure that the user is able to input and upload the form in the expected manner. Test cases will also test for exceptions: 1) User misses a required entry, 2) User inputs invalid entry 3) System shutdown during data upload.
004	5.2	Validate the browsing and search functions. Test cases will include scenarios for software not able to connect to database and search query yielding no results.
005	5.3.1	Validate the delayed upload feature for specific form selected by user. Test cases will include tests for exceptional scenarios covered in SRS to ensure that the software behaves in an expected manner.
006	5.3.2	Validate the batch delayed upload feature. Test cases will include tests for exceptional scenarios covered in SRS to ensure that the software behaves in an expected manner.

7.5 APPROACH

Component Testing	Components will be tested by test cases to ensure they are working during the software development cycle.
Integration Testing	A detailed set of test cases will be created and then executed. These test cases will cover all the requirements detailed in the SRS.
Interface Testing	To ensure that all aspects of the display are rendered correctly, the external interfaces regarding data entry forms and browsing and searching will be tested manually by using the user interface to perform all product tasks.
Security Testing	Security of the software will be tested by making attempts to access the data without proper authentication.
Performance Testing	Test users will interact with the system to ensure that the system and its search and browsing utility respond in a reasonable time.

7.6 PASS/FAIL CRITERIA

Suspension Criteria	Test case execution will be suspended in the event of discovery of a critical failure that affects the value of the tests.
Resumption Criteria	Once the developer thinks the problem causing the suspension has been resolved, test case may resume.
Approval Criteria	The results for each test case will be considered approved if the results meet the expected results description in the test case.

7.7 SCHEDULE

Task	**Duration**
Develop test cases	11/12 - 11/24
Develop scripts for automated testing	11/24 - 12/3
Execute tests	12/3 - 12/8
Report defects	12/3 - 12/8
Complete test report	12/8 - 12/11

8 FUTURE EXTENSIONS

- User Voice Interface
- Expand location
- Smart watch application/feature
- Additional OS Support

Software is currently localized to two specific towns. The application can be extended to cover more towns and locations in the future. Search and browse functions in the software currently supports keypad and mouse/finger interface. Future extension can include a user voice interface as well. Software can be extended to possibly include full support as a smartwatch application. Software currently supports Android OS. It can support other operating systems in the future.

9 APPENDIX A. Example Screens

Include copies of specifications, mockups, prototypes, etc. supplied or derived from the customer. Appendices are labeled A, B, . . . n. Reference each appendix as appropriate in the text of the document.

[insert appendix A here]