**Data Management and The Foodel System**

Neil Urquhart,

School Of Computing

Edinburgh Napier University

2/6/21

*n.urquhart@napier.ac.uk*

**Introduction**

This document is intended to provide a brief description as to how data is processed and stored by the Foodel system. Foodel is an application developed by Dr Neil Urquhart of Edinburgh Napier University, it is designed to help voluntary organisations and small businesses with last-mile delivery problems.

**Foodel Architecture**

Foodel is designed to be as simple as possible to use and configure. The optimisation software (the component that builds routes) runs as a web server, this server may be on the users’ machine or it could be on a shared server allowing wider access.

In configurations where sensitive data is being processed, it is recommended that Foodel is run locally and only accessed from that machine. By default Foodel uses port 8080 to communicate via Http with the web browser. This port can be changed from within the ./config/server.properties file supplied with Foodel.

*When Foodel is being used locally, it is recommended that a firewall be used to block remote access to the port used by Foodel.*

If access is granted to the Foodel port from other machines, then care should be taken to enable keys (see below).

When a problem is uploaded to Foodel a unique *key* is generated and displayed to the user, in order to make Foodel more secure the use of keys may be enabled (adding “nokeys=false” to ./config/server.properties), in this mode the user will have to enter the “key” in order to examine a problem solution. This option should be enabled if shared access to the server is possible.

**Foodel Data and Storage**

A standard Foodel problem file requires the following data for each visit to be made

Name The id of the recipient (can be any valid string of chars)

Address/postcode A postcode (e.g. EH10 5DT) or a complete address.

Demand An integer number (default 1) which defines the size of the order

Order Note (Optional) Free text that will be displayed on the drivers’ schedule next to the delivery

In addition problems may also contain volunteer data, to allow for volunteer scheduling,

Volunteer Name The id of the volunteer (can be any valid string of chars)

Volunteer Address A valid address (or postcode) which denotes the volunteers’ start and end point

By default data is **NOT** stored on disk by the server, it is held in memory whilst the server is running. The user can use the Print function to print a copy of the result or to save it as a .PDF.

The system may be configured to save a copy of each problem/solution in JSON format in the .\sols\ folder if ‘savefiles=true’ is enabled within the server.properties file. Please be aware that JSON is an easily readable format and is not encrypted.

If the ‘savefiles’ option is set to false then all data and solutions are lost when the Foodel software is terminated.

**Data Processing**

The principle data processing carried out within Foodel is as follows:

* Address/postcode strings are geocoded to longitude and latitude using a connection to the nomination.org server . These pairings are not cached on the server.
* Address/Postcode strings are grouped into routes and sorted into delivery order
* Recipient and Volunteer information is overlaid onto map data for display

**Recommendations**

1. Do not enable savefiles unless you have a specific reason to wish to retain access to the solutions.
2. If savefiles is enabled, please ensure that access rights to the sols folder are protected
3. Please ensure that the port used by Foodel for communicating with the web browser is protected by a firewall.
4. If access is provided to Foodel from hosts other land local host then enable keys

A reasonably secure configuration for single-user use on a “local” machine would be:

* Port 8080 blocked by firewall software for remote connections
* Server.properties to contain the following:

nokey=true

savesolutions=false