# 

NASA ISS FIT Administration Tool Architecture and Deployment Guide

|  |  |  |
| --- | --- | --- |
| **Author** | **Revision Number** | **Date** |
| pvmagacho | 1.0 | 04/17/2015 |
| pvmagacho | 1.1 | 04/29/2015 |
| pvmagacho | 1.2 | 06/03/2015 |
| pvmagacho | 1.3 | 08/07/2015 |
| pvmagacho | 1.4 | 08/13/2015 |

[NASA ISS FIT Administration Tool Architecture and Deployment Guide 1](#_Toc421103192)

[Architecture 3](#_Toc421103193)

[1. Overview 3](#_Toc421103194)

[1.1 Application Requirements 3](#_Toc421103195)

[1.2 Application Overview 4](#_Toc421103196)

[1.3 Application Structure 4](#_Toc421103197)

[1.4 Application Configuration 4](#_Toc421103198)

[Deployment Guide 5](#_Toc421103199)

[2. Application Deployment 5](#_Toc421103200)

[3. Application Verification 8](#_Toc421103201)

# Architecture

# Overview

The customer of this project is the NASA Nutritional Biochemistry Laboratory working on tracking dietary intake of astronauts on the International Space Station (ISS). The NASA Nutritional Biochemistry Laboratory collects the data on type and quantity of food eaten by astronauts on the ISS, and uses these for medical reporting as well as for scientific research.

NASA has been using a special Food Frequency Questionnaire (FFQ) for tracking weekly consumption of food by astronauts. It is a long list of food products grouped by several categories according to their nutrient content. Each astronaut filled in that questionnaire with the quantity of food packages that he/she consumed during the week. The FFQ approach allowed astronauts to collect dietary intake data for a week.

The Laboratory performs offline analysis of that data found in the database, determines nutrient components consumed by each astronaut, and searches for various relationships between food diet data and physical conditions/performance of space crew. A prompt feedback on the calculated results can be given to the Flight Surgeon and the crew member (this feature is out of scope for the application, but is the ultimate goal of the data coming from the application).

The main goal of this project is to deliver a solution for efficient collection of food consumption information by astronauts on the ISS. It should provide precise, fast, and easy to use tracking ability for every day usage in micro-gravity conditions on the ISS. Convenient features for user identification, preparing food lists, various ways to input information and data synchronization with the Earth laboratory are also important parts of this project.

This module implements the administration tool, which will control the information found in the database and generate reports.

## Application Requirements

* Debian 6 (Squeeze) Linux
* Node.js
* Express
* Jade
* pg-format
* python-shell
* Postgresql 9
* JQuery
* JQuery Datatables

## Application Overview

The IFIT Admin Tool is built using NodeJs and Express to handle incoming HTTP calls.

Data is retrieved from a postgresql database installed in the same computer running the admin tool.

Jade templates are used to render the retrieved information from the database as HTML pages.

Reports are generated from the tool and can have information from one or many users. The output file is an excel file with the food consumption information.

To generate the reports python scripts are used to connect to the Postgresql database and retrieve all needed information.

## Application Structure

* **Admin Tool/database/** - folder with the database schema
* **Admin Tool/public/** - folder with images, css and javascript files used by web pages
* **Admin Tool/views/** - folder with Jade templates
* **Admin Tool/reports/** - output folder for reports generated from the tool
* **Admin Tool/csv bulk upload templates/** - folder with CSV template files to be used with bulk upload tool feature
* **Admin Tool/** - folder with the main app.js, package.json node configuration file and python scripts

## Application Configuration

The configuration is done directly inside the app.js. The database name, username, password and hostname can be found in the variable conString for app.js. The default value is:

**"postgres://postgres:12345@localhost/nasadb"**

* Username: postgres
* Password: 12345
* Hostname: localhost
* Database: nasadb

The python scripts have the database name, username and password as arguments. The hostname is hardcoded as localhost (127.0.0.1).

# Deployment Guide

# Application Deployment

The deployment should be made in a Debian 6 (Squeeze) linux distribution.

The following steps should be followed to setup the Admin Tool.

1. Install Debian 6 (Squeeze) linux – **skip if already installed**. The link to download the distro is <https://www.debian.org/releases/squeeze/debian-installer>
2. Login as **root** user
3. Run the following commands in prompt

> apt-get -y install wget build-essential git-core curl openssl libssl-dev sudo

1. Install postgresql 9 – **skip if already installed**

> sh -c 'echo "deb http://apt.postgresql.org/pub/repos/apt/ squeeze-pgdg main" > /etc/apt/sources.list.d/pgdg.list'

> wget --quiet -O - https://www.postgresql.org/media/keys/ACCC4CF8.asc | sudo apt-key add -

> apt-get update

> apt-get upgrade

> apt-get -y install postgresql-9.0 libpq5 libpq-dev postgresql-contrib-9.0

1. Install nodejs

> git clone https://github.com/joyent/node.git

> cd node

> git checkout v0.10.29

> ./configure --openssl-libpath=/usr/lib/ssl

> make && make install

1. Install python dev

> apt-get -y install python-dev python-psycopg2

1. Configure postgresql database
   1. Open file /etc/postgresql/9.0/main/pg\_hba.conf and add the following line at the end

**host pl\_fit pl\_fit\_db 0.0.0.0/0 md5**

* 1. Open file /etc/postgresql/9.0/main/postgresql.conf and modify the following line:

**listen\_addresses = '\*' # what IP address(es) to listen on;**

**port = 56283 # (change requires restart)**

**ssl = false # (change requires restart)**

* 1. Restart server

> service postgresql restart

1. Create NASA database

> su -l postgres

> createdb pl\_fit

> psql -d pl\_fit

pl\_fit=# ALTER USER postgres WITH PASSWORD '12345';

pl\_fit=# CREATE USER pl\_fit\_db WITH NOSUPERUSER NOCREATEDB NOCREATEROLE ENCRYPTED PASSWORD 'CHANGEME';

pl\_fit=# \q

> exit

1. Configure Admin tool (**online machine**)

> npm -g install node-gyp

> git clone https://github.com/nasa/NTL-ISS-Food-Intake-Tracker.git

> cd NTL-ISS-Food-Intake-Tracker/Admin\ Tool

> npm install

> cd database

> psql -h localhost -d pl\_fit -W -U postgres -p 56283 -f database\_schema.sql

(when prompt enter password: 12345)

> cd ../

1. Configure Admin tool (**offline machine**)

> Download code repository zip file (<https://github.com/NASA-Tournament-Lab/NTL-ISS-Food-Intake-Tracker/archive/master.zip>) and copy to offline machine

> unzip NTL-ISS-Food-Intake-Tracker-master.zip

> cd NTL-ISS-Food-Intake-Tracker-master/Admin\ Tool

> cd database

> psql -h localhost -d pl\_fit -W -U postgres -p 56283 -f database\_schema.sql

(when prompt enter password: 12345)

> cd ../

1. Configure login credentials

> su -l postgres

> cd /usr/share/postgresql/9.0/contrib

> psql -d pl\_fit

pl\_fit=# \i pgcrypto.sql;

pl\_fit=# INSERT INTO users(username, email, pwdhash) values('pl\_fit\_db', 'pl\_fit\_db@email.com', crypt('CHANGEME', gen\_salt('bf')));

pl\_fit=# \q

> exit

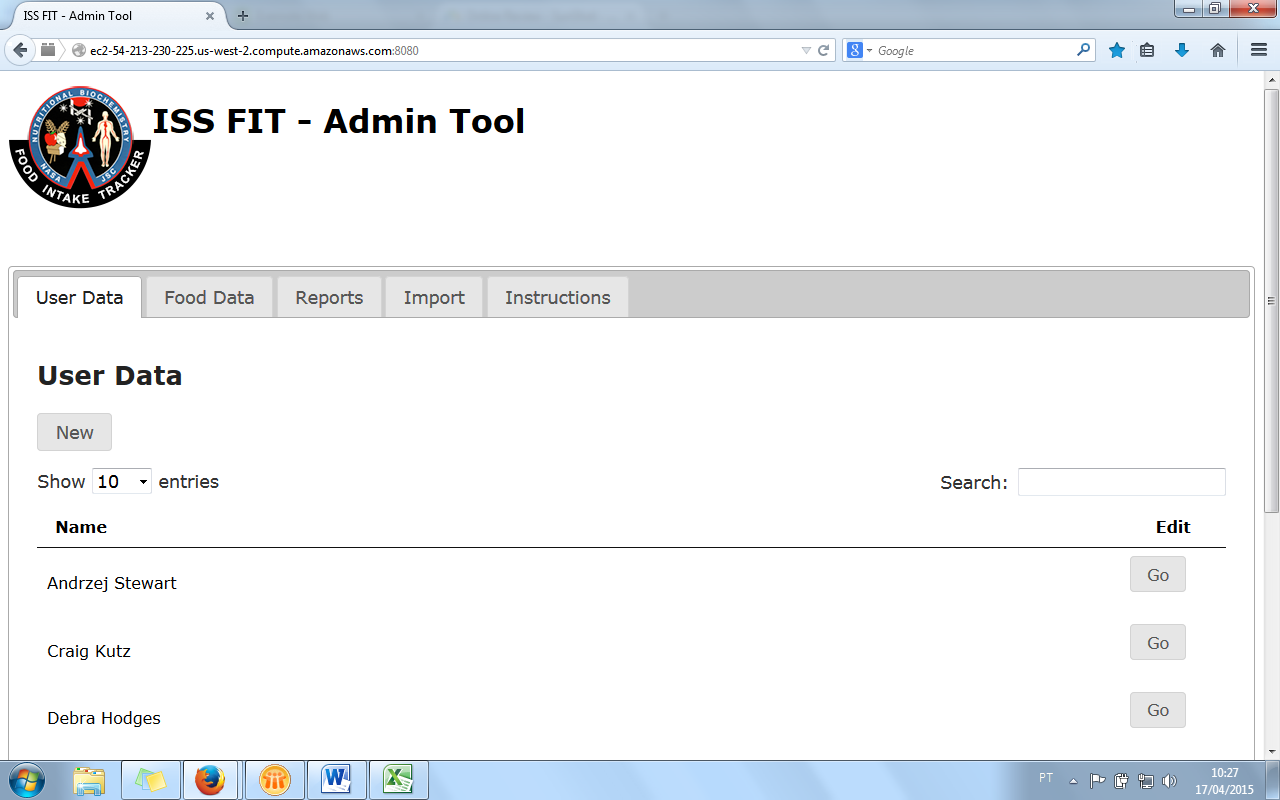
1. Start application

> node app.js

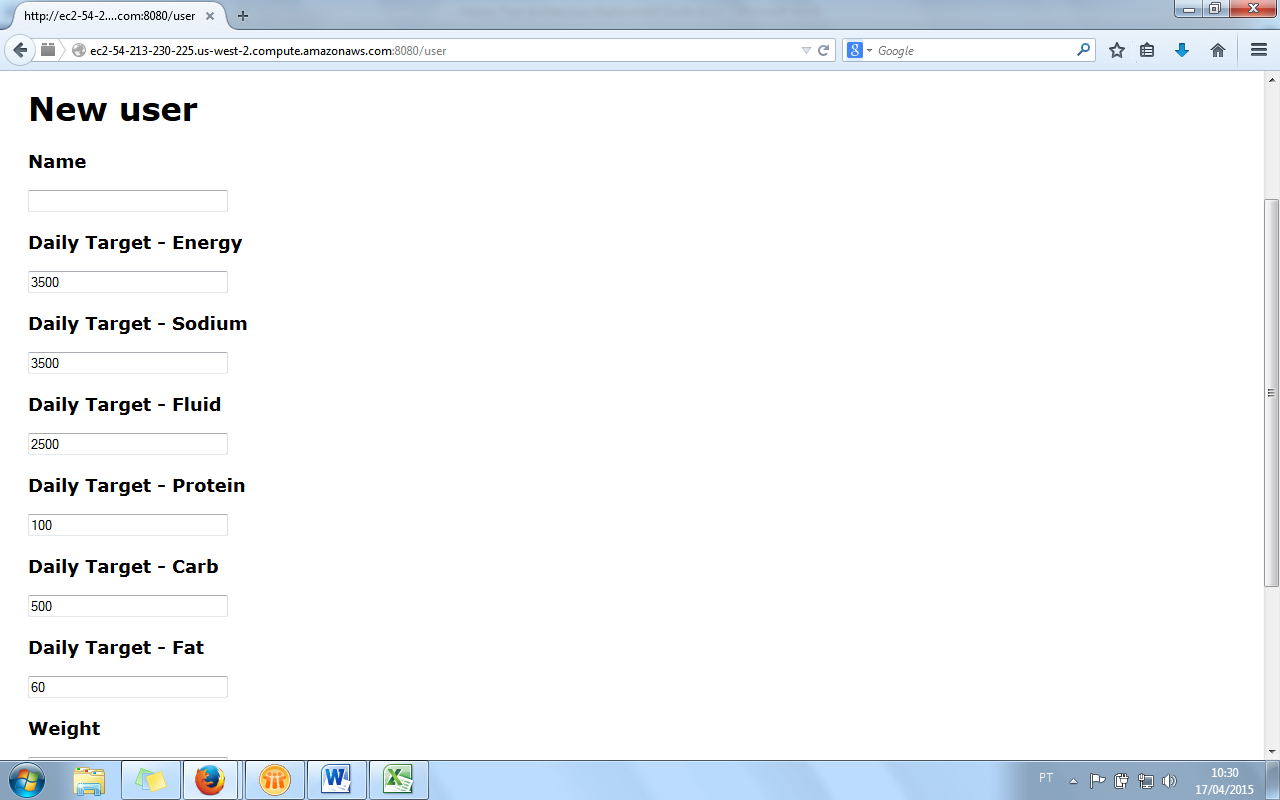
# Application Verification

To verify the application open the browser with the IP of the server and port 8080.

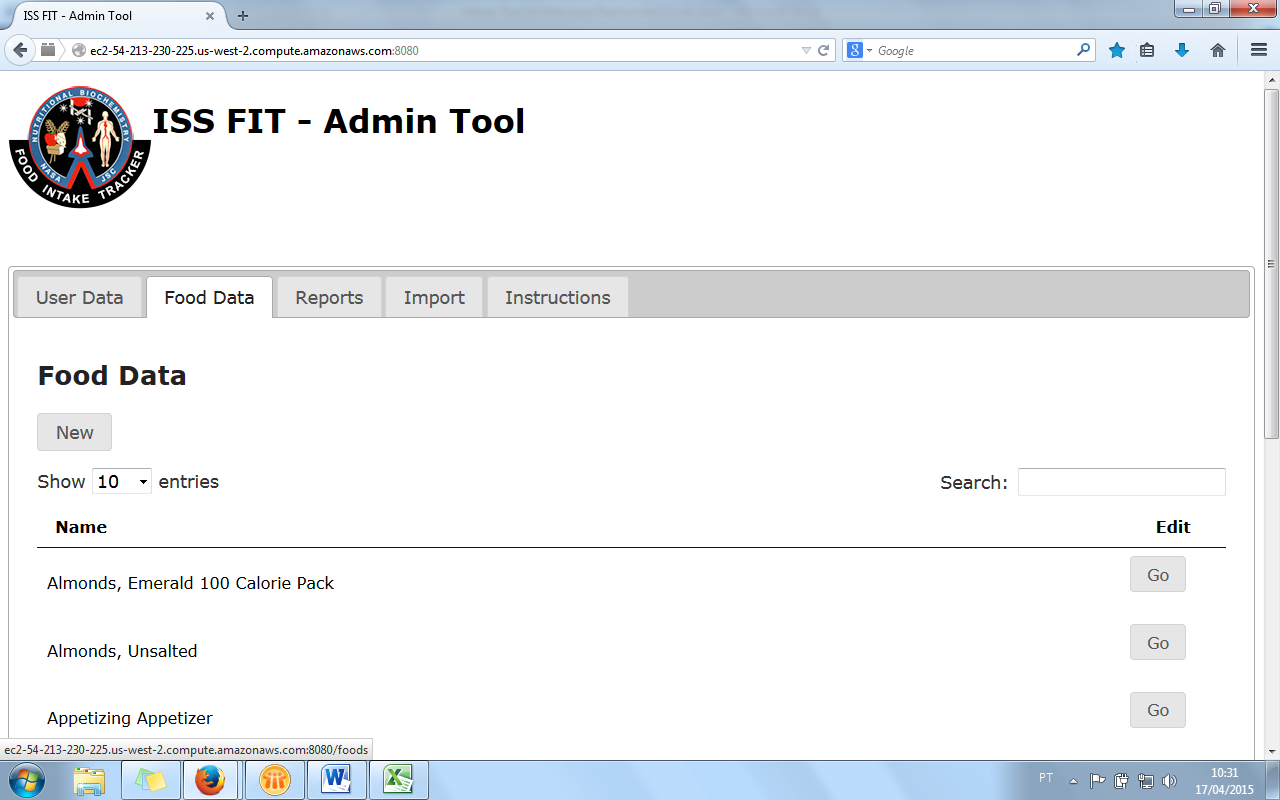
* http://{SERVER\_IP}:8080

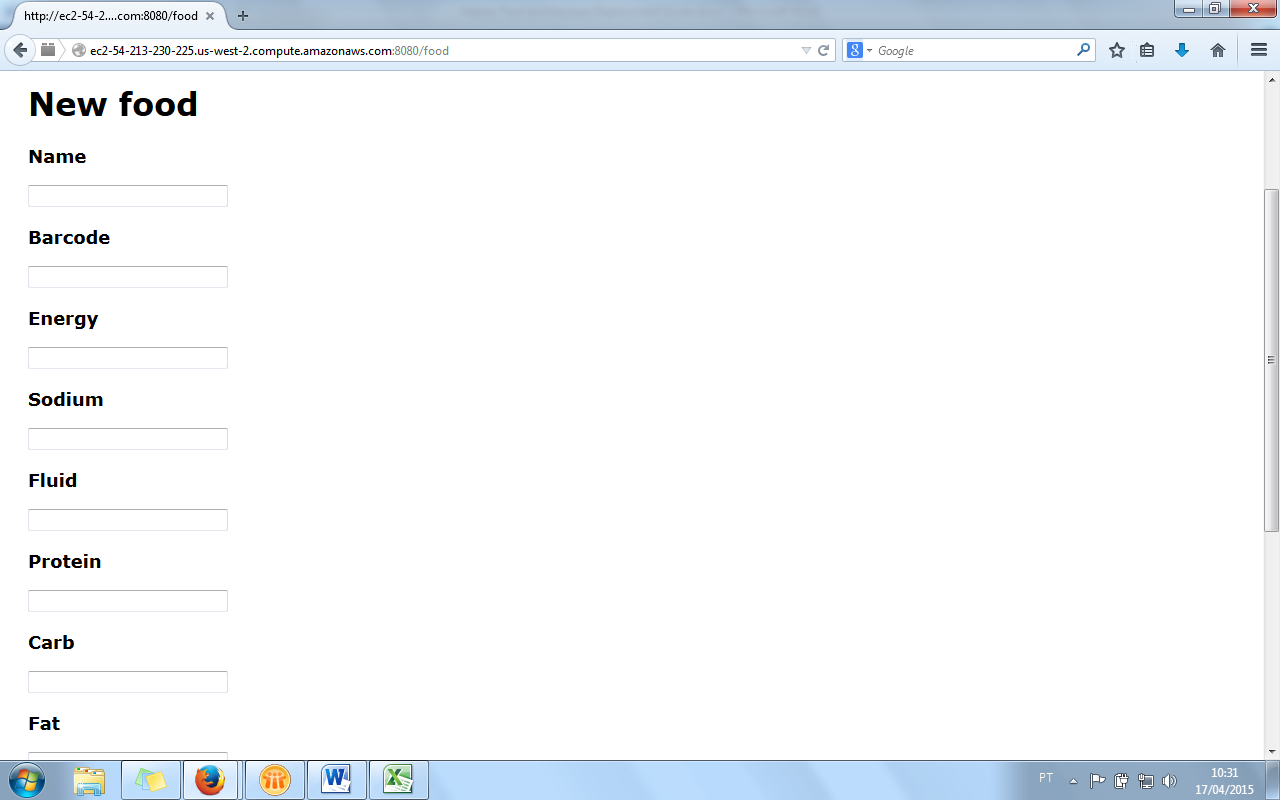


* Click New to add new user.

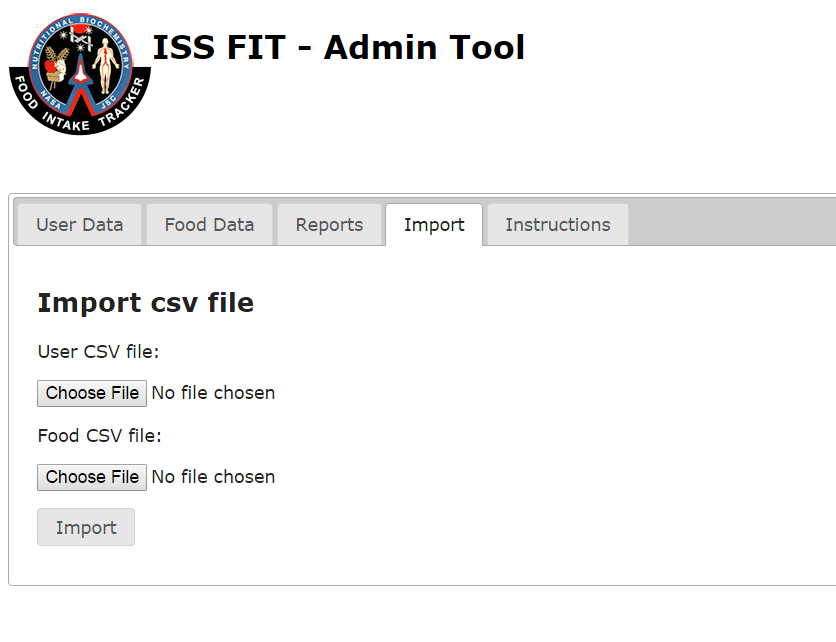


* Click “Food Data” tab. Click New to add new food.





* Click “Import” tab to load CSV files. Instructions can be found in “Instructions Tab”



* Click “Reports” tab to generate reports. Select users or all users’ option.

