**CSE3OAD/CSE4OAD - Assignment 2**

**Due Date: 10 amMonday 16 October 2017**

**Assessment:** This assignment 2 is worth 25% of the final mark for CSE3OAD/CSE4OAD.

This is an individual assignment.

**Copying, Plagiarism:** Plagiarism is the submission of somebody else’s work in a manner that gives the impression that the work is your own. The Department of Computer Science and Computer Engineering treats plagiarism very seriously. When it is detected, penalties are strictly imposed.Students are referred to the Department of Computer Science and Computer Engineering's Handbook and policy documents with regard to plagiarism.

**No extensions will be given:** Penalties are applied to late assignments (5% of total assignment mark given is deducted per day, accepted up to 5 days after the due date only). If there are circumstances that prevent the assignment being submitted on time, an application for special consideration may be made. See the departmental Student Handbook for details. Note that delays caused by computer downtime cannot be accepted as a valid reason for a late submission without penalty. Students must plan their work to allow for both scheduled and unscheduled downtime.

***Assignments submitted more than 5 days late will receive the mark of 0.***

**Return of Assignments:** Students are referred to the departmental Student Handbook for details.

**Objectives:** To implement web services and develop web application in AngularJS.

**Introduction**

The aims of the assignment are:

* To implement web services for making available resources regarding a collection of recipes. The recipes are those that you have worked with in assignment 1.
* To implement a basic single-page application in AngularJS and to make use of those web services to build this single-page application.

**Files Provided**

The following are provided in the source/src directory (of the zip file)

* The jar files for JSON conversion, the servlet-api jar, and MySQL driver (in directory lib)
* CreateRecipeDatabase.sql

You can adapt this file to your own need. However, the table nams and definitions should remain unchanged as this will effect our ability to mark your assignment using our database.

* The class Recipe.java
* The class RecipeDSC.java
* The class RecipeWS (To be COMPLETED)

The following are provided in the source/web directory.

* Completed
  + index.html
  + Recipe.js, Ingredient.js
* To be completed
  + main.html, display.html. add.html, edit.html
  + main.js

**Task 1 – Building Web Services**

You are required to build the web services to

* Get all the recipes
  + Parameter: None
  + Data Sent (to server): None
  + Data Received (from server): JSON array representing all the recipes
* Get the recipe of a given id
  + Parameter: an id (to be attached to the URL)
  + Data Sent: None
  + Data Received: JSON object representing the recipe
* Add a recipe
  + Parameter: None
  + Data Sent: JSON object representing the recipe
  + Data Received: JSON object representing the recipe
* Edit (update) a recipe
  + Parameter: an id (to be attached to the URL)
  + Data Sent: JSON object representing the edited recipe
  + Data Received: JSON object representing the newly updated recipe
* Delete the recipe of a given id
  + Parameter: an id (to be attached to the URL)
  + Data Sent: None
  + Data Received: None

The URL for all requests should be: http://localhost:8080/Assignment2/ RecipeWS

This should be configurable from the DBSettings.txt file.

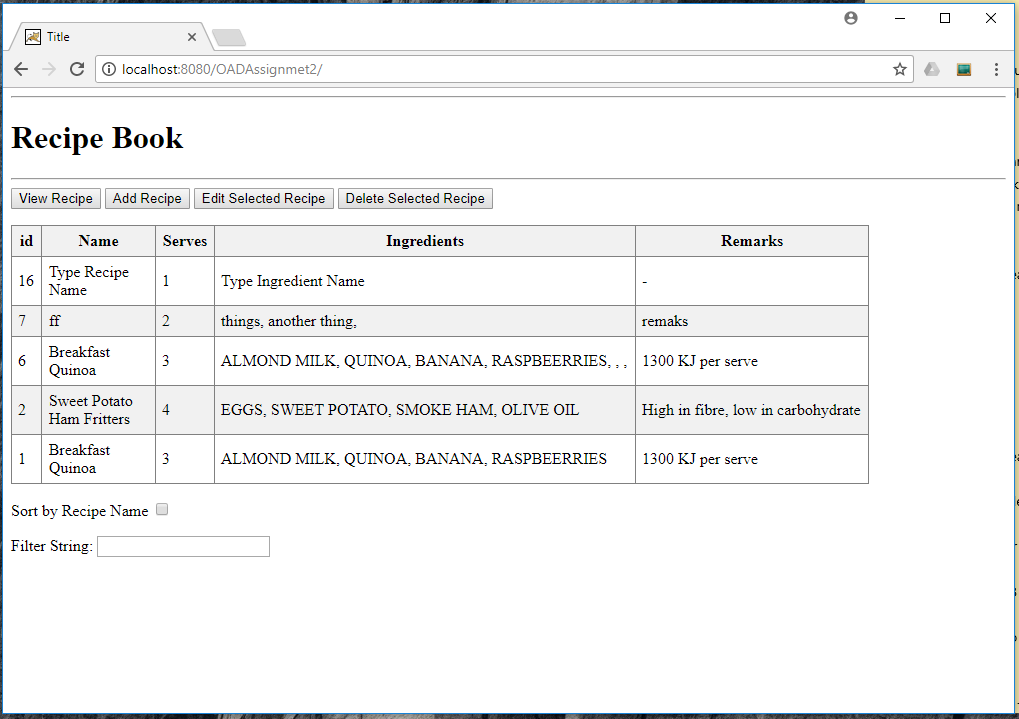
**Task 2 – Implementing an AngularJS Application**

You are required to build a basic single-page application. More specifically, the application provides

* A main page (main.html). A page to list all the recipes and allow the user to delete them
* A page to display a recipe (display.html)
* A page to add a recipe (add.html)
* A page to edit a recipe (edit.html)
* All of these are driven by one controller (in main.js using Ingredinet.js and Recipy.js)

**Displaying the main page (main.html)**

This page displays a page like this.



This page displays the list of all the recipes. For each recipe, it displays all the details, except the steps. (The steps of a recipe can be displayed by display.html.)

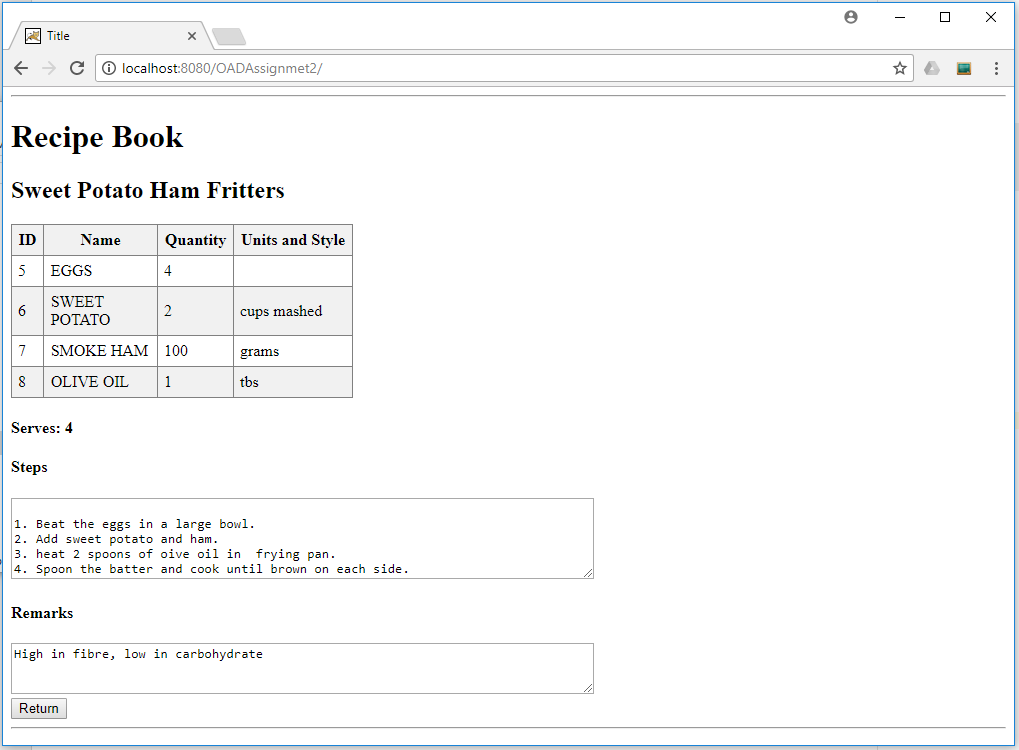
The check box should toggle between ordering by name and ID.

The filer string can be used to filter the recipes. A recipe is displayed if any one of its fields contains the string in the text field. If the filter string is empty, it means no filtering and all the recipes are displayed. The matching is not case-sensitive. An illustration is shown below.

From this page the user should be able to delete a recipe. There should be a conformation dialog but no other addition UI page is needed.

**Displaying a recipe (display.html)**

Should produce a page that looks like this if the user has selected a recipe to be displayed

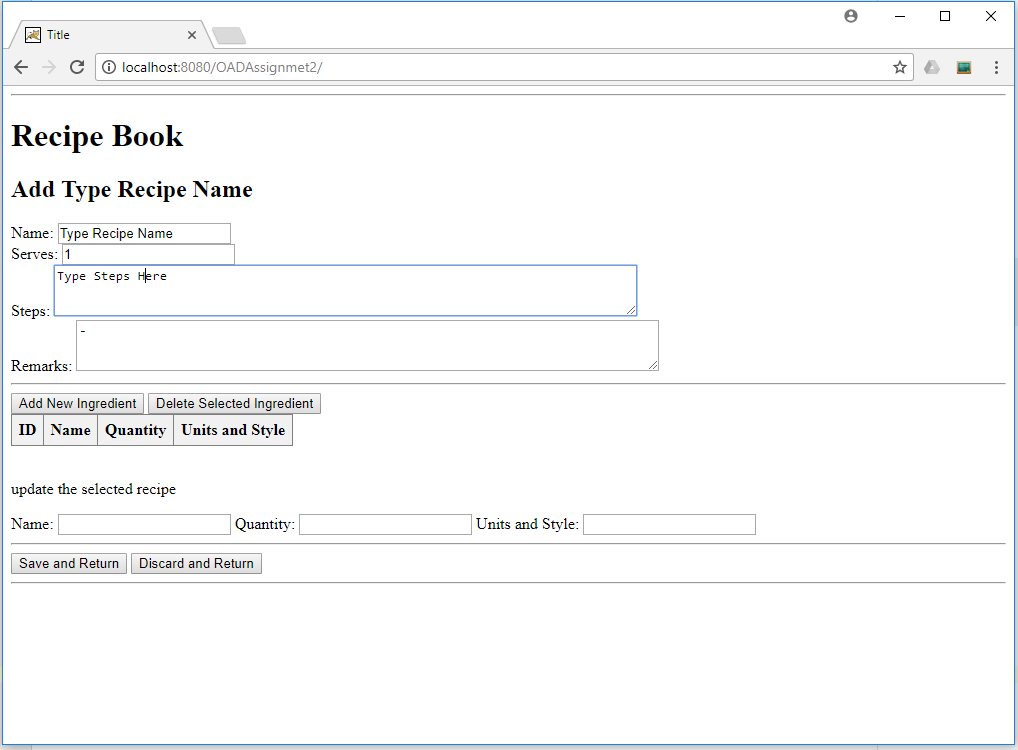


Uses h2, h4, and textarea with readonly.

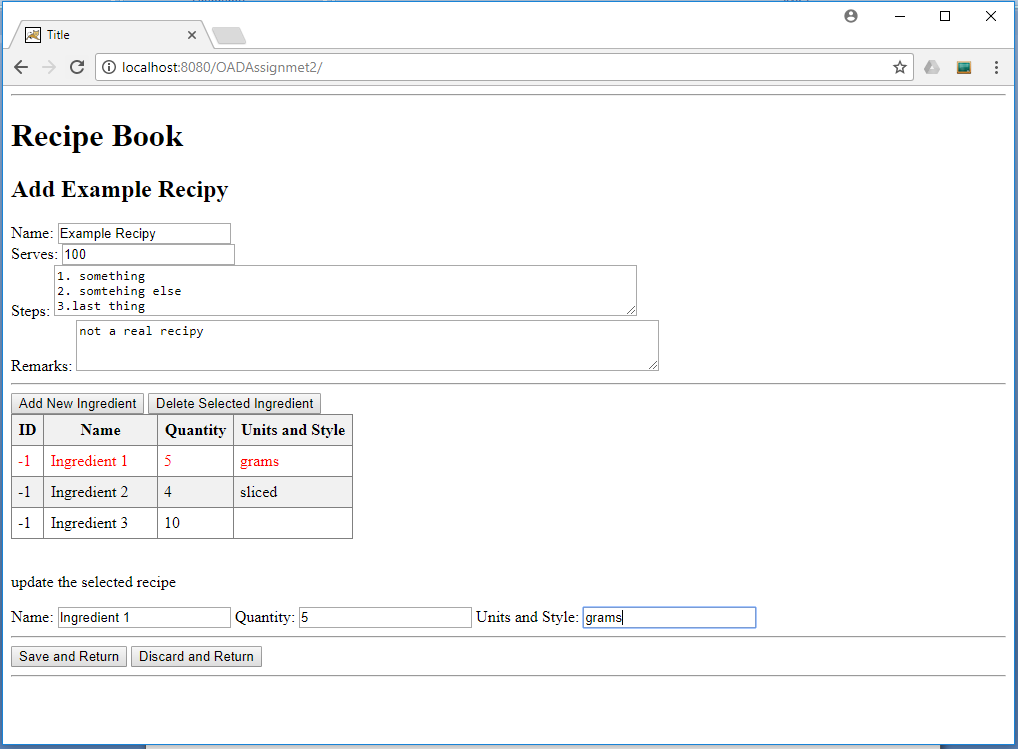
The text area can be set to have the display area of 5 rows and 80 columns, for example.

**Adding a recipe (add.html and add.js)**

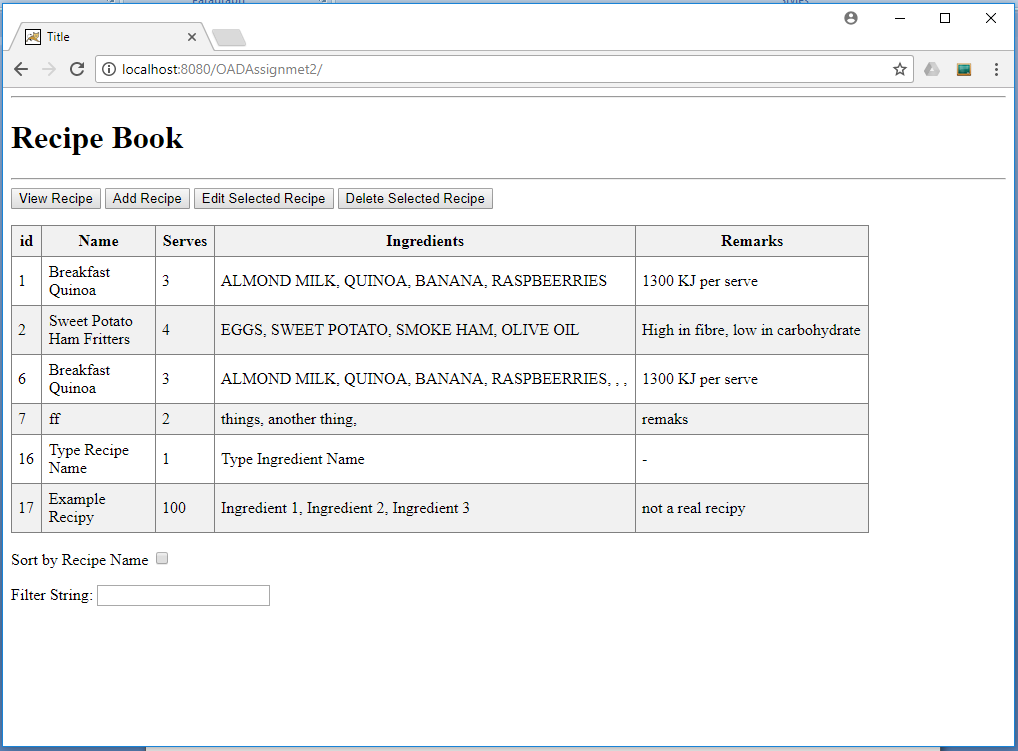
This page shows a form for the user to enter details of a new recipe. Note that there are no entry field for id, and initially there are initially no ingredients to show



Once you start adding ingredients they should be added to the table



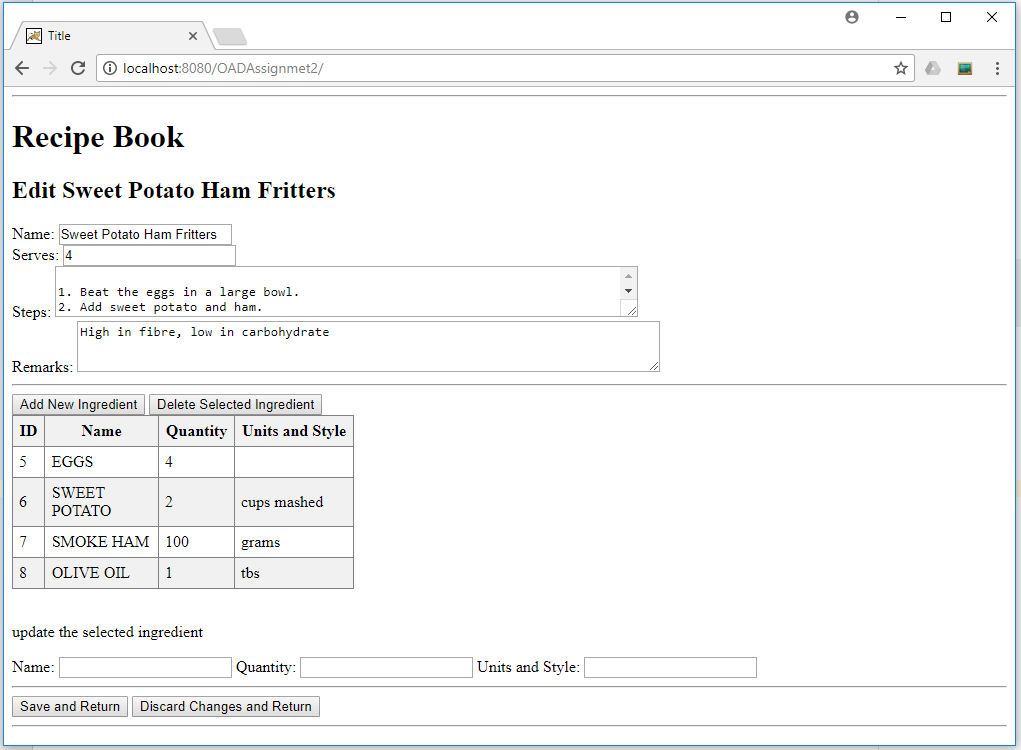
If there are no errors in saving the recipe then the page should return to the main page where the user can see that their new recipe has been added



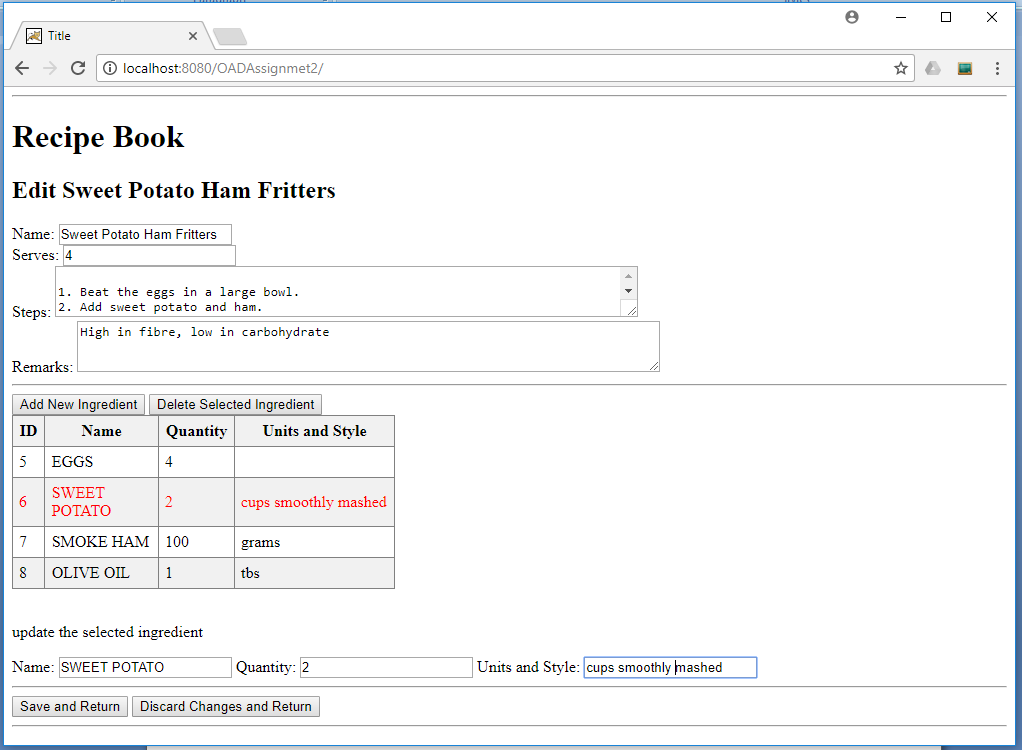
**Editing a recipe (edit.html and edit.js)**

Displayes the selected recipe as shown below.

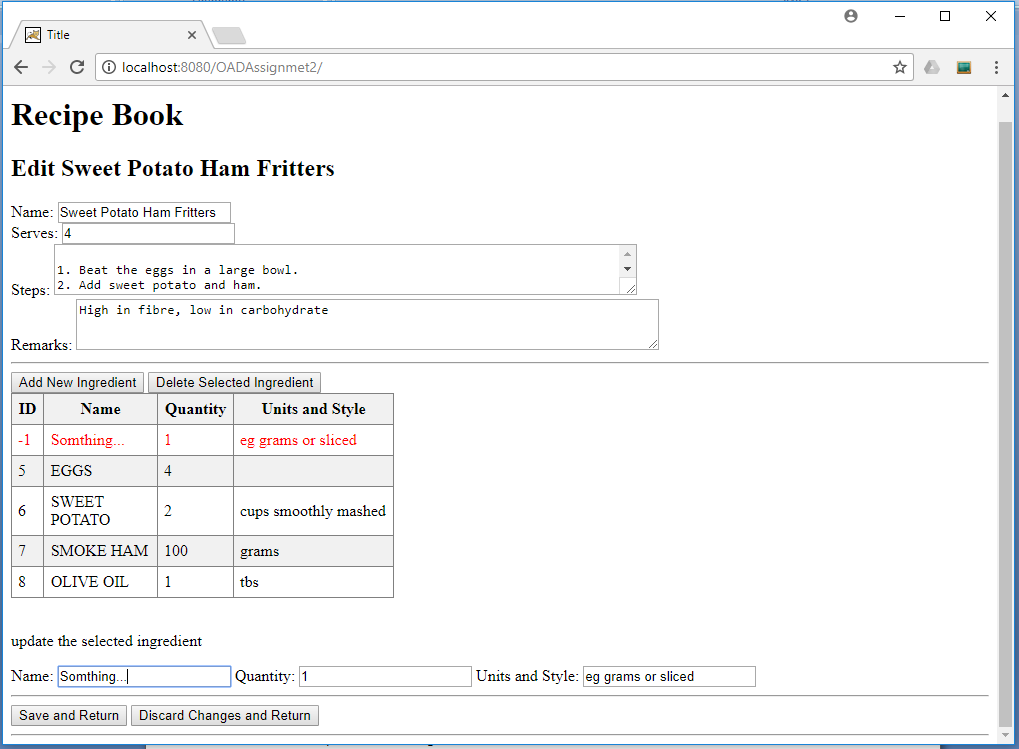
Each ofthe fields should alow the user to edit the various values



When the user selects an ingredient they should be able to edit it using the fields below ‘update the selected ingredient’



Clicking Add New Ingredient should add a new default ingredient to the table and select it so that the user can update the new ingredients values



**What to submit**

* Electronic copy of all of the classes required to run the application, *including those that are provided*, are to be submitted to latcs8using the submit OAD <directory or filename>command

***Note that the submission is not through LMS.***

* Within your submission should be 2 folders.
  + A web services source containing all your java source files (including the provided files) and a lib folder with the provided jar files. All of the classes that you created or completed must be able to be compiled from their current directory. This means, they must not be contained in any package.
  + Then a second folder containing all your html and java script files, including all provided files.
* As for the database, you can use the one on latcs7 or on your local machine. The only requirement is that your program should work on the table recipe which must have the same structure as the one in the provided MySQL script.
* The database URL, username, and password must be configurable via the DBSettings.txt file.
* For classes that you create/alter, you must include, as part of the comments, your name, and student ID.

***Assignments without this will have 5% of the mark deducted.***

Note: a marking rubric will be released soon to LMS.