University of Zurich[UZH]

Department of Informatics

University of Zürich
Department of Informatics
Binzmühlestr. 14
CH-8050 Zürich
Phone. +41 44 635 43 11
Fax +41 44 635 68 09
www.ifi.uzh.ch/dbtg

UZH, Dept. of Informatics, Binzmühlestr. 14, CH-8050 Zürich

Prof. Dr. Michael Böhlen
Professor
Phone +41 44 635 43 33
Fax +41 44 635 68 09
boehlen@ifi.uzh.ch

Zürich, 17. Januar 2017

Software Project: Development of a dynamic web application

CONTEXT

The Swiss Feed Database[1] contains historical data about nutrient content of animal feeds. It is used by stock-farmers, feed mills, etc. to find out dietary quality of various animal feeds and so determine healthy, effective, and cheap diets for livestock. A web application is provided that allows a user to define a set of selection criteria (e.g., nutrient and feed types, temporal and geographical information about feed samples), retrieve nutrient content data, and overview results in various intuitive forms. The goal of this project is to develop a second version of this application following the Model-view-controller design pattern that clearly separates data management (model), interface (view), and business logic (controller) of the application. The interface of the application will also be redesigned to be more user friendly (e.g., screen size must be adjusted to user's screen). Finally, data transfer will also be considered so that the new version of the application be more responsive. For example, in the current implementation all matched rows are retrieved even if the client browser displays only a part of them. Methodologies that retrieve each time only the needed part of data must be considered.

The Swiss Feed Database contains data for about 1000 feed types and 900 nutrients. The major part of nutrients are grouped according to the animal species that they are relevant to, and according to the chemical category (e.g., amino acids or mineral). These groups are used in the application to simplify navigation for the end user. Also, for each sample temporal and geographical provenance information are stored.

The Feed Database provides two views on the containment of nutrients in the animal feeds: detailed and summary. Detailed view allows to investigate nutrients of individual samples. Individual samples can be chosen based on location, time, feed and nutrient names. Summary

[1] `http://feedbase.ch`

view provides a reference value that quantifies the containment at the current date for each feed type and nutrient. Reference values are highly demanded by farmers and feed industry which are not interested in the analyses but in forming good diets for their livestock. The Swiss Feed Database offers extended access rights to subscribed users.

FUNCTIONALITY

The web application must provide a login form and a task bar providing search options for queries. The buttons of the task bar must be loaded dynamically depending on data stored in the database, the access rights of the user and already chosen options. The task bar must also provide two search modes. Summary data search (reference values) with limited search options (only feed type and nutrient) and sample (detail) data with extended search options (geographical and temporal provenance information). The workflow of creating a query should be the following:

1. Select summary or detail search mode.
2. A list of available feed types is loaded from the database and a selection menu is provided to the user.
3. After selecting some feed types, a similar menu with available nutrients (related to the selected feed types) is loaded and a selection menu is provided.
4. If detailed data is selected similar menus are provided to restrict the time and location that feed samples are collected.
5. Once a query has been formed, it can be saved and retrieved later from a list of saved queries.

After a query has been formed and posed, data are presented to the user by three different report forms.

Sortable table A table with one nutrient per column and one feed type per row must be generated. Row number is limited (e.g., 50 rows at most) and the user can display next (previous) 50 rows by pressing right (left) arrows. A click into any column heading sorts the table in ascending or descending order. Table report form is enabled both for summary and detailed data.

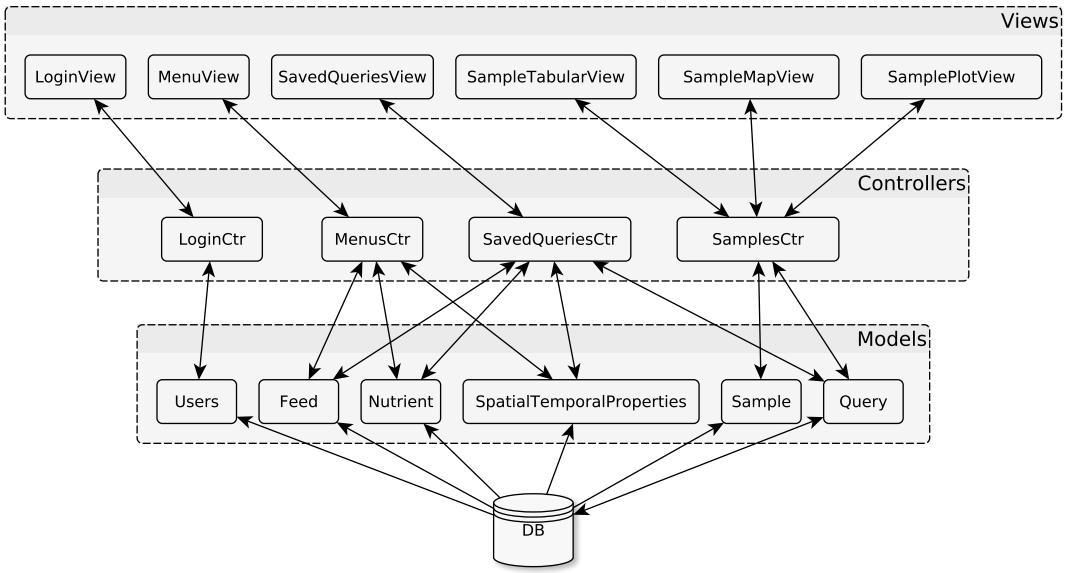
Scatter plot Variation patterns of along the time are of particular interest. Thus, a scatter plot is provided that visualizes the nutrient content (y-axis) of retrieved samples as a function of time (x-axis). Drop down menus enable a user to select nutrients to be plotted.

Sample origins map A map depicts the origin location of samples.

In case of the sortable table, only the 50 first rows are presented each time but the client first retrieves and parses all rows matching the search criteria and splits them in 50-row pages. This approach makes the application less responsive sometimes. In case of a rapid implementation, the remaining time can be used to develop methodologies that make the application more responsive by avoiding to load more data than needed or by generating and loading representative data if possible.

ARCHITECTURE

The suggested architecture of the project is depicted in the following figure:



Models define objects (e.g., User, Feed) representing data in the database and provide basic operations on them (e.g., retrieve all available feeds). Model objects are available to controllers so that they implement the actual business logic of the application. For example, LoginCtr must implement an authentication process proving that a user is registered in the database and retrieving the user's access rights. Finally, views design the application interface. For example, LoginView must provide a simple login form, while MenuView must provide a dynamic menu for selecting feed types, nutrients, etc.

MILESTONES

Milestone 1: Overview of the current situation (1 week)

The schema of the Swiss Feed Database and the current web application implementation must be studied and the new implementation must be designed (providing UML diagrams).

Milestone 2: Main functionality (1 week)

The main functionality must be implemented that should include (i) a login form, (ii) a query task bar that provides menus, dynamically loaded, to form queries, (iii) capability to save a query and retrieve a list of all saved queries.

Milestone 3: Samples retrieval and basic visualization (1 week)

The functionality for applying a query (probably saved), retrieving the matching samples and presenting them in a sortable table must be implemented.

Milestone 4: Advanced Visualization (2 weeks)

The visualization forms of a scatter plot and a map must be implemented. For this purpose it is suggested that libraries of Google Maps and Google Visualization are used.

Milestone 5: Documentation (1 week)

A technical documentation of the implementation must also be prepared. The delivered documentation must be oriented to developers. The architecture of the application must be presented along with descriptions about implemented models, views, and controllers. Instructions to install the application and running examples must also be provided.

Start date: 20.01.2016
End date: 03.03.2016

Supervisor: Georgios Garmpis (ggarmpis@ifi.uzh.ch)

University of Zurich
Department of Informatics

Prof. Dr. Michael Böhlen
Professor