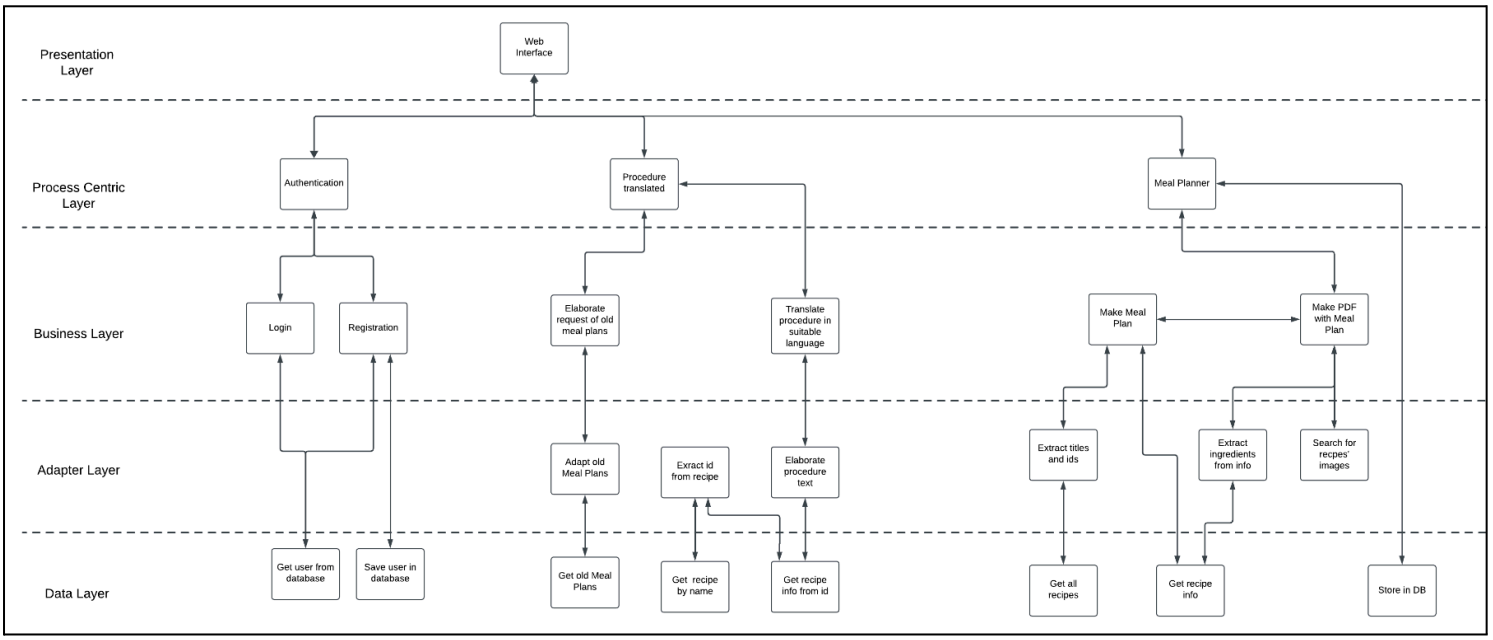
Meal Plans service

Service Design and Engineering final project, 27/01/2025

Gabriele Volani (257019)
Salvatore Gilles Cassarà (257820)

Repository: https://github.com/gabri17/sde-project

Introduction



Our project has been developed with a **python** service-oriented application and our services are provided with a **web interface**.

We are using an online **MongoDB** cluster instance as a database, for saving information about *users* and *meal plans*.

All the 24 services involved in the project communicate internally with a series of **API calls** and are divided into 4 typology:

- **Data Services Layer**: 9 services involved in operation with the database and asking data from external providers.
- **Adapter Services Layer**: 5 services involved in filtering and transforming data provided by the previous layer, tailoring it to meet our requirements.
- **Business Logic Services Layer**: 7 services involved in manipulating our data, elaborating and processing the information to perform the needed computations.

- **Process Centric Services Layer**: 3 services managing the requests from the user interface and offering the specific service, by orchestrating and composing together services of the previous layers.

The **external** services are:

1. Spoonacular API: a free web service offering access to a vast database of recipes, with lots of information associated with.
2. DuckDuckGO Search API: a web service used to perform an image search on the Internet.
3. DeepL API: a web service that allows translations of text in different languages.

Our project is made up of three different main Process Centric Services:

1. Meal Planner
2. Procedure translator
3. Authentication

Authentication process

Authentication is simply made up with 2 services at the business layer and are used in order to provide **login** and **registration** functionalities, interacting with Data Layer services.

Security aspects are managed following the OAUTH 2.0 standard: after doing login a temporary **JWT** token is generated and sent back to the client, where it will be stored in a cookie.

Then it will be included as a query parameter for the API requests that need authentication.

Meal planner process centric service

Going into the details of the **Meal Planner**, it is used to create a daily meal plan, which is made up of a recipe for lunch and one for dinner. It behaves as follows:

- The user accesses this process by using the “Create Meal Plan” button from the Web Interface;
- The user is asked to select its dietary constraints (if any);
- These constraints are passed down to the Data Layer, which uses them to request 100 recipes that satisfy these constraints from the Spoonacular API;
- All the recipes are passed to an adapter in the Adapter Layer that extracts Title and Id of each recipe from the JSON returned by Spoonacular;
- The list of 100 recipes is then passed to the Business Layer, where 2 of them are randomly picked;
- The 2 selected recipes go back to the Data Layer and their Ids are used to request at the Spoonacular API the full details list of those recipes;
- The details are passed to the Adapter Layer, where the list of ingredients is extracted from the JSON;
- The list of ingredients and recipe names is then passed to the Business Layer, but to make the final PDF of the meal plan, we still miss the images;

    - The images are requested by a service in the Adapter Layer using the DuckDuckGO Search API to look for images based on the recipe titles and are then sent to the Business Layer;
- The PDF can now be created using an internal API;
- Finally, the created PDF is passed to the Process Centric Layer, which returns it as a FileResponse to the Web Interface in the Presentation Layer, where the file is downloaded on the user's device.

Procedure translator process centric service

In this process centric service, the user provides a recipe name and gets back the procedure to follow in order to realize it, translated in the desired language.
Let's go in the detail in the workflow:

1. user needs to sign in into the application using the login service;
2. user get all the history on the meal plans created for him by database and select one specific recipe;
3. user provides the recipe name and the target language to the service;
4. recipe name arrived at the Data Layer where the Spoonacular API is called in order to some basic information of the recipe;
5. thanks to an Adapter service, the id of the recipe is retrieved;
6. with another Data Layer service, another Spoonacular API is used to get the full details of the recipe;
7. an Adapter service manipulate the object retrieved and elaborate the procedure of the recipe;
8. a Business Layer service process the object by translating it into the desired language, using the DeepL API;
9. an object with the translation of the procedure and the title of the recipe is sent back to the user.