ASSIGNMENT OF BACHELOR’S THESIS

Title: CookBook - menu planning application
Student: Alexey Karpovskiy
Supervisor: Ing. Zdeněk Rybola, Ph.D.
Study Programme: Informatics
Study Branch: Web and Software Engineering
Department: Department of Software Engineering
Validity: Until the end of winter semester 2020/21

Instructions

The goal of this thesis is to create an application for menu management in a restaurant. The main features are: recipe management, stock management, menu preparation and approval, statistics of meals.

Instructions:
- Analyze the CookBook application created during the Software Team Project
- Analyze similar existing solutions
- In cooperation with the supervisor, suggest and analyze improvements of the application
- Design the realization of the improvements
- Realize the designed improvements and test the solution
- Update the documentation of the system and prepare the release

References

Will be provided by the supervisor.

Ing. Michal Valenta, Ph.D.
Head of Department
doc. RNDr. Ing. Marcel Jiřina, Ph.D.
Dean

Prague February 18, 2019
Bachelor’s thesis

**CookBook - Menu Planning Application**

*Alexey Karpovskiy Bachelor*

Supervisor: Zdenek Rybola

25th June 2019
Acknowledgements

I would like to thank my supervisor Ing. Zdeněk Rybola, Ph.D. for supporting me with the topic and for all his valuable advices during the creation of the thesis.

Above all, I would like to thank my family for giving me all the opportunities I had and for all the support during my studies.
I hereby declare that the presented thesis is my own work and that I have cited all sources of information in accordance with the Guideline for adhering to ethical principles when elaborating an academic final thesis.

I acknowledge that my thesis is subject to the rights and obligations stipulated by the Act No. 121/2000 Coll., the Copyright Act, as amended, in particular that the Czech Technical University in Prague has the right to conclude a license agreement on the utilization of this thesis as school work under the provisions of Article 60(1) of the Act.

In Prague on 25th June 2019

Declaration
Czech Technical University in Prague
Faculty of Information Technology
© 2019 Alexey Karpovskiy. All rights reserved.
This thesis is school work as defined by Copyright Act of the Czech Republic.
It has been submitted at Czech Technical University in Prague, Faculty of
Information Technology. The thesis is protected by the Copyright Act and its
usage without author’s permission is prohibited (with exceptions defined by the
Copyright Act).

Citation of this thesis
Karpovskiy, Alexey. *CookBook - Menu Planning Application*. Bachelor’s
thesis. Czech Technical University in Prague, Faculty of Information Techno-
logy, 2019.
Abstrakt

Daná diplomová práce je zaměřena na řešení plánování menu s funkcemi pro správu receptů a ingrediencí. To zahrnuje analýzu aplikace po předmětu BIE-SP2, návrh a analyzu rozšíření aplikace, návrh a realizace těchto rozšíření, včetně otestování, přípravu dokumentace systému a přípravu vydání finální verze aplikace.

Klíčová slova Restaurace, Plánování menu, Správa recepturou, Správa ingrediencí, Informační systémy.

Abstract

This bachelor thesis aims to create the solution for menu planning with recipe management and ingredient management features. This includes the analysis of the CookBook application after the BIE-SP2 subject, suggestion and analysis of improvements, design the realization of improvements, test the solution, prepare the documentation of the system and prepare release of the final version of the application.

Keywords Restaurants, Menu planning, Recipe management, Ingredient management, Information systems
## Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Introduction</strong></td>
<td>1</td>
</tr>
<tr>
<td>1. Goal of the Thesis</td>
<td>2</td>
</tr>
<tr>
<td>2. Structure</td>
<td>2</td>
</tr>
<tr>
<td>1 Analysis</td>
<td>5</td>
</tr>
<tr>
<td>1.1 Current Version</td>
<td>5</td>
</tr>
<tr>
<td>1.2 New Requirements</td>
<td>8</td>
</tr>
<tr>
<td>1.3 Domain</td>
<td>11</td>
</tr>
<tr>
<td>1.4 Existing Solutions</td>
<td>13</td>
</tr>
<tr>
<td>2 Design</td>
<td>15</td>
</tr>
<tr>
<td>2.1 Technologies</td>
<td>15</td>
</tr>
<tr>
<td>2.2 Architecture</td>
<td>18</td>
</tr>
<tr>
<td>2.3 Database</td>
<td>19</td>
</tr>
<tr>
<td>2.4 Class model</td>
<td>22</td>
</tr>
<tr>
<td>2.5 Sequence model</td>
<td>23</td>
</tr>
<tr>
<td>3 Implementation</td>
<td>27</td>
</tr>
<tr>
<td>3.1 Tools</td>
<td>27</td>
</tr>
<tr>
<td>3.2 Implementation Details</td>
<td>28</td>
</tr>
<tr>
<td>3.3 Testing</td>
<td>30</td>
</tr>
<tr>
<td>Conclusion</td>
<td>33</td>
</tr>
<tr>
<td>Bibliography</td>
<td>35</td>
</tr>
<tr>
<td>A Acronyms</td>
<td>37</td>
</tr>
<tr>
<td>B Contents of enclosed CD</td>
<td>39</td>
</tr>
</tbody>
</table>
List of Figures

1.1 Current Domain Model ........................................... 7
1.2 New Domain Model ............................................... 12
1.3 Request State Machine ....................................... 12
1.4 Requesting Process Model .................................. 13

2.1 Deployment Model ............................................... 19
2.2 Architecture ..................................................... 20
2.3 Database Schema ............................................... 21
2.4 Menu Controller Class ....................................... 22
2.5 Main View Class ............................................... 23
2.6 Add Recipe Sequence Model ................................ 24
2.7 Get Overview Sequence Model .............................. 26
Restaurants is a complex business consisting of many processes. Every day restaurants encounter a lot of different problems which should be solved very quickly otherwise they slows the work down. This is why every process should be organized and controlled in order to reduce the number of causes of problems.

One of those processes is cooking, the most integral part of the restaurant business. This process should be organized first. Moreover, cooking consists of several different processes. First is menu planning. The cook should prepare the menu for every day in a week which also contains the planned number of portions for every dish. Since every restaurant specializes on the different cuisines, this involves the recipe planning. Some of the dishes are not ordered often or should be replaced for different reasons, the menu also changes.

The process of recipe creation starts another one, the ingredients planning. Every recipe contains the ingredients needed and it’s amount. The ingredients which will be used during the whole week should be supplied and stored to the stock. Moreover, the number of ingredients in stock also should be taken into account. But the situation in reality often shows that can happens that there is no enough ingredients since they were not supplied or simply spoiled.

When all the plannings are finished, the menu and ingredients supply should be discussed with the manager. It can happen, that the menu is declined by the manager and must be planned again.

Taking into account all the details above, it was decided to create the system which will help in organization and control of the cooking process.
Introduction

Goal of the Thesis

The goal of this thesis is to create an application for menu management in a restaurant with the following main features:

- To manage meal recipes with required ingredients
- To manage the days menu consisting of several meals with expected number of served portions
- To create an approval process of the proposed days menu by restaurant management
- To manage the ingredients stock and supply it on daily basis
- To note statistics of served portions to support the planning of days menu portion numbers

For completion of the goal, the thesis includes the analysis of the latest version of the CookBook application after the BIE-SP2 subject, detection of problems and not covered initial requirements, improvements and suggestion of improvements of the application, analysis of existing solutions in order to become competitive on the market. All the requirements discussed during the thesis should be implemented in order to release the final version of the application. The installation package and instructions also should be prepared for being provided to the potential customers.

Structure

This thesis includes the following structure:

- Analysis. This section discusses the analysis of the current version of the CookBook application, new requirements, domain of the application and analysis of the existing solutions.
  - Current Version. In this section the current state of the application, initial requirements and the existing problems in the current version are discussed.
  - New requirements. Based on the analysis in this section discussed initial requirements which were not complete and improving requirements which were suggested and discussed with the supervisor.
  - Domain. In this section discussed the new domain model which compared to the initial requirements and domain model of the current version of the CookBook application.
Structure

- Existing solutions. This section includes the analysis of the existing solution on the market and comparison to the CookBook application.

- Design. This section describes the technologies used in the development process, architecture of the application, database model, class and sequential models.
  - Technologies. In this section discussed the stack of the technologies used during the development process.
  - Architecture. In this section discussed the architecture of the CookBook application.
  - Database. In this section discussed the database of the application.
  - Class model. This section contains the presentation and discussion of the samples of class models the application contains.
  - Sequence model. This section contains the presentation and discussion of the samples of sequence models of the application.

- Implementation. This section contains the discussion of the tools used during the development process, implementation details and tests.
  - Tools. In this section discussed the list of tools used during the development process.
  - Implementation details. This section contains nontrivial and interesting moments during the implementation process.
  - Testing. In this section discussed tests of the CookBook application.

- Conclusion. This section contains the conclusion of the thesis.
1.1 Current Version

In this section the current version of the CookBook application is discussed. It contains the requirements which were implemented, the domain model of the current version and the description of the problems.

The initial version of the CookBook application was created during BIE-SP1 and BIE-SP2 subjects by the group of 9 students: Nilay Baranwal, Giorgi Khazaradze, Toghrul Sultanzade, Rishabh Malik, Bektsultan Baatyrbekov, Abdullah Abdullah, Alexey Karpovskiy, Alina Vigriyanova, Giorgi Khazaradze.

1.1.1 Requirements

The current version of the CookBook application does not satisfy the initial requirements which were planned at the requirement stage of the project. The requirements which were implemented in the CookBook:

- Show Recipes
  Shows the list of recipes currently saved in the system

- Add Recipe
  New recipe can be created using existing ingredients from stock. The amount of ingredients can be specified.

- Update Recipe
  The recipe can be updated, however it contains only the update of ingredients functionality.

- Show Menus
  Shows the list of menus currently saved in the system.

- Add Menu
  The menu for the specific date can be created based on the list of recipes
existing in the system. The number of portions planned and served can be specified.

- Update Menu
  The list of dishes in the menu can be changed and the updated menu is saved in the system.

- Approve/Decline Menu
  Menu can be approved or declined by the manager after review.

- Ask for an approve of the Menu
  The request for approve can be created by cook at the step of the menu creation.

- Evaluate Menu (Comment section)
  At the step of the menu reviewing comments from the manager or cook can be created.

- Show Stock
  Shows the list of ingredients which currently stored in stock.

- Add Ingredients to Stock
  The process of adding new ingredient to stock.

The non-functional requirements which were implemented in the CookBook:

- Web application
- The web application availability

1.1.2 Domain Model of the current version

On the [1.1] is shown the Domain Model of the current version of the application. As it can be seen from the model, it does not fulfil the objectives faced for application. The main entity in this model is Menu, which is a composition of Comments and Dishes. Cook and Manager are the realisation of User. Cook creates and update Menu, Manager reviews the Menu. The Dish has the recipe which contains the Recipe how to cook this dish. Every Recipe contains the Ingredients in Recipe which has the Ingredient used. However, Recipe is missing several important attributes, such as Instructions, Category and if it can be used for cooking Vegetarian dish.

All the Ingredients are the Stock Item. But the Stock Item is missing the attribute Expiration Date.

Finally, there is no possibility to create a request to provide Ingredient to Stock. This functionality is included in the initial requirements of the application.
1.1.3 Application

The current version of CookBook application contains bags on the steps of creation and modification of menus. It is not possible to add several new dishes to menu because it always causes the Null Pointer Exceptions. Another known bag is a duplication in database when manager approves or decline the menu. The same problems were discovered with ingredients which are used in recipes at the creation dishes step.
1. Analysis

The reason of those problems is the current database schema. Since in the current version of the application is used the schema which is auto-generated, the mistakes in association mappings simply generated the wrong database schema which is caused the unexpected behavior. This problem was not discovered on the step of implementation because this part of the functionality was not tested properly.

Moreover, the part of the functionality such as creation and modification of dishes in menu was implemented based on this schema, it should be re-implemented for new database schema.

Also, several views in the current version of the application need to be changed in order to become more user friendly and intuitive.

1.2 New Requirements

As it was discussed above, the database schema is the cause of bugs. Since the already implemented functionality is based on the current schema with the creation of new schema the current functionality also needs to be modified or re-implemented. The functionality needed to fulfil the main business processes should be implemented.

1.2.1 Functional requirements

In order to complete the pursued goals the new requirements need to be implemented. First of all, the missing requirements that were set at the initial stage:

- FR1 Request of new Ingredients
  It should be possible to create a request for missing ingredients since the menus in restaurants are usually planned in advance.

- FR2 Apply Menu for Manager review
  The logic around menu creation should be changed. Now every change in the menu automatically resubmits the menu to the Manager for review.

- FR3 Show Recipe Details in Menu
  The Cook can use the application to see what is the menu for current date. It should be possible for him to get instruction how to cook a dish and the amount of the ingredients needed for specified amount of portions.

The current version of CookBook application is missing any flexibility in operations with data. One of the reasons is difficulty to modify already added dishes in the menu, ingredients in menu and stock ingredients. The following delete functionality is required:
1.2. New Requirements

- FR4 Delete Menu
  The menus which are not in use anymore or rejected should be possible to delete.

- FR5 Delete Recipe
  The recipe which is irrelevant can be deleted from the system.

- FR6 Delete Ingredients in Recipe
  It was not possible to delete an ingredient from recipe if it was added by mistake.

- FR7 Delete Dish in Menu
  The same problem with the dishes in menu. Added by mistake dished cannot be deleted.

- FR8 Delete Ingredients in Stock. This functionality is needed for stock managements. When the expiration date logic will be added the stock will contain also the expired ingredients which should be deleted.

Now the Stock is just a list of added ingredients which contains only basic information about ingredients we use and the quantity. Since one of the CookBook applications goals is to help manage the Stock, all the Ingredients should have the expiration date and logic around it. That is why the following requirements around the expiration date should be implemented:

- FR9 Show Expired Ingredients in Stock
  This functionality should ease the stock management. It selects and shows all the expired items so that they can be easily deleted.

Also, should be modified one of the existing functional requirements which work with the Stock:

- FR10 Add Ingredients to the Stock
  This functionality with the expiration date added to Ingredient requires modification of whole Stock Item class. Especially, the comparison and creation.

One of the goals of the application is to show the information about the Menu and Ingredients in stock so that the Cook and Manager can see the missing ingredients, they can keep further Menu discussion or can be created a request for missing ingredients. So this will require the ingredients overview functionality:

- FR11 Show Ingredients Overview in Menu
  In the menu this functionality is included in order to get information about the amount of ingredients left in stock in order to see if more portions can be cooked. In restaurants, when the daily menu is approved
1. Analysis

and the number of portions planned, can happen that the real amount of portions is higher.

- FR12 Show Ingredients Overview for chosen period of time
  This functionality is needed in order to get summary on the ingredients usage by menus. This includes the information for a day, week or month because all the menus in restaurants planned in advance. With this functionality can be created the request on missing ingredients.

The possibility to get the Overview make possible to apply changes directly to the database at the end of the day.

- FR13 Apply Ingredients Usage to the Stock
  Manager can get the overview of used ingredients for a chosen day. This functionality will apply changes to the stock. The real usage of the ingredient can be modified then this number is applied. Once the ingredient usages applied to stock the menu is closed and saved for statistic.

1.2.2 UI changes

New requirements such as Overview and Requests require the views.

- VR1 Create View for Overview functionality
- VR2 Create View for Requests functionality

When the application will be used in the restaurants there will be a big amount of data which now is not filtered at all. The most useful functionality is possibility to search by name.

- VR3 Search Ingredients
- VR4 Search Recipes
- VR5 Search Dishes

With the modification of Recipe Entity and adding the cooking instructions, the Views which responsible for the creation or modification of Recipe should now include corresponding field.

- VR6 Modify View for adding new Recipe
- VR7 Modify View for editing the Recipe

Also, the Menu details view should contain the possibility to show instructions in front of every dish it contain to make it easier for Cook to follow the process.

- VR8 Modify View for showing Menu details
1.3. Domain

Since the functional requirements mentioned above contain changes in Stock with addition of Expiration Date there should be created new view for adding new stock ingredient.

- **VR9 Modify View for adding Ingredient to Stock**

Now the process of adding new Ingredients to the Recipe is made in two steps. First - add Ingredients in one View, second - modify quantity in another. To simplify the process of creation new Recipes the new view for adding ingredients should be created and contain only one step.

- **VR10 Modify View for adding Ingredient in Recipe**

The same should be done when adding new Dishes to Menu

- **VR11 Modify View for adding Dishes in Menu**

Current version of application contains the bunch of pop-up windows for managing content which make many processes uncomfortable since they simply cant contain all needed information and buttons. All the main pop-ups should be transformed to the full size windows in order to make an application more user friendly.

1.3 Domain

New domain model 1.2 is now covers all the requirements of the CookBook application described above. Requests for ingredients now exists and the states of requests included. With addition of new attributes the application now cover most of the real needs as the restaurants menu planning tool.

The attributes added in the new domain model ease to the Cook menu creation process and actual cooking process since now the Recipes have instructions for the Cook. Also, the Recipes now contains the categories such as, for instance: main, soup, salads and deserts. This allows to filter the recipes for dishes during the menu creation process. Moreover some dishes can be marked as Vegetarian since this attribute is included in Recipe.

Another important attribute is an Expiration Date for a Stock Item. It allows to store same Ingredient in stock with the different expiration date.

The following state machine 1.3 demonstrates the Request states. When the Request is created it is stated as Opened. Once Manager decides, that the Ingredients in the request should be bought, he can approve it and the Request gets the status Approved. If Manager thinks that the Request should not be accepted, he can simply close it and the Request is stated as Closed. Only the Approved request can be considered as an active.

The Requesting Activity Diagram is demonstrated 1.4
1. Analysis

Figure 1.2: New Domain Model

Figure 1.3: Request State Machine
1.4 Existing Solutions

Nowadays, there are two notable companies that provide menu management tool. The first one is JAMIX (www.jamix.com). A Finnish restaurant management software provider, specialising in applications for professional kitchens, staff restaurants. And another one is PARSLEY (www.parsleysoftware.com) which is providing the smart software for chefs.

The solutions that these companies provide have the same functionalities and both are paid. They both have a recipe planner, menu planner and the stock management tool. Even the features of those functionalities are the same. For instance, the nutrition facts of every meal and the cost of every
1. Analysis

ingredient. It is really difficult to differ between them.

There are a lot of data needed to be filled in in the software from Jamix and Parsley. For instance, it requires to fill every nutrition fact. This makes the application overflow with the information. Undoubtedly, this is an important information but it is often taken as a formality.

The CookBook application also has the main functionalities such as recipe, menu and stock management which already makes it competitive. The stock features such as, for instance, request for missing ingredients and expiration date will be also implemented. But what makes the CookBook application better then the compared software is that it is lightweight, not overflowing with extra formal information and it is free to use.
2.1 Technologies

This section contains the technologies which are used during the development process and the reasons why exactly these technologies are chosen.

2.1.1 Programming Language

The programming language that is used in CookBook application is Java.[1] There are four main reasons why this language was chosen for the development of CookBook.

- Extensibility
  Java is simply extensible for web app development. If there will be a need in scaling of the application, Java language is acceptable for those requirements. It makes this language really comfortable for reaching the long term goals.

- Frameworks
  There are a lot of frameworks available for Java web development which make the development process faster and easier. Moreover, every popular framework has a huge amount of all needed documentation, support and community.

- Platform Independent
  There is no need in a particular platform. Java is a great example of code portability.

- Garbage Collection
  Java can easily handle garbage data, if the pile is filled, then the garbage is collected and destroyed. Which again makes this language appropriate, for instance, in terms of application scaling process.
2. **Design**

2.1.2 **Frameworks and Technologies**

The frameworks and technologies which were used in the CookBook application.

- **Hibernate**
  
  Hibernate is a free, open source object-relational mapping library for Java designed to map objects to an RDBMS and to implement the object-oriented programming concepts in a relational database. Hibernate provides support to JPA annotations like @Entity, @Table, @Column, etc. These annotations make the code portable to other ORM frameworks. Unlike JDBC, Hibernate connects with the database itself and uses HQL (Hibernate Query Language) to execute the queries, then maps the results to Java objects. All of that simplifies the implementation of Data Layer, that is why this framework was chosen.

- **Spring**
  
  The Spring framework is a powerful enterprise Java framework. It was designed to simplify Java EE development and make development more productive. Framework which is used in the CookBook application is Spring Boot. The main goal of Spring Boot Framework is to reduce Development, Unit Test and Integration Test time and to ease the development of Production ready web applications very easily compared to existing Spring Framework, which really takes more time.

  Reasons why the Spring Boot framework is used in the development:

  - Spring Boot uses Hibernate as the default JPA implementation.
  - Embed Tomcat, Jetty or Undertow directly (no need to deploy WAR files)
  - Provide opinionated 'starter' dependencies to simplify build configuration
  - Automatically configure Spring and 3rd party libraries whenever possible

  All these advantages of the Spring Boot framework allows to create a simple application just in couple of minutes which can be evolved in more complex.

- **Vaadin**
  
  The presentation layer is implemented with the help of Vaadin framework. This framework is good for building UIs for business web apps without having to deal with multiple front-end technologies. In practice, Vaadin framework helps to reduce development time when building the
2.1. Technologies

UIs using web technologies. Also, it allows to build web applications in one language (Java) and ensure web-security. This is why all the Views of the CookBook application are implemented using this technology.

2.1.3 Database

MySQL is a freely available open source Relational Database Management System (RDBMS) that uses Structured Query Language. MySQL was designed and optimized for Web applications. As new and different requirements emerged with the Internet, MySQL became the platform of choice for web developers, and the default database for web-based applications. This is why it is used in the CookBook application.

2.1.4 Web Server

Apache Tomcat is an open source web server and servlet container designed to execute Java Servlets and render web pages that use Java Server Page (JSP) coding. Tomcat is an incredibly lightweight application. It offers only the most basic functionality necessary to run a server, meaning it provides relatively quick load and redeploy times. It is an extremely stable platform to build on and using it to run applications will contribute to servers stability, as well. In despite of all the pros of Tomcat, its free distributed what makes it a great choise for CookBook application.

2.1.5 PaaS

Platform as a Service (PaaS) is a cloud computing model that helps with speedy and accurate application development. It is a system in which a third party provider delivers these app development tools via a internet connection, with the necessary hardware and software supporting it hosted off-site. With a PaaS, everything what is needed in a platform is managed: operating systems, server software, firewall and security systems.

One clear advantage to using PaaS is that the only spendings is cloud computing. Its saving money that can be used for other business operations. There is not need to maintain, upgrade, or replace systems and software. With a PaaS, test and development of application speeds up.

In the CookBook application this technology was used for deployment of test and sample versions of the application.

2.1.6 DBaaS

Database as a service (DBaaS) is a cloud computing service model that provide users with some form of access to a database without the need for setting up physical hardware, installing software or configuring for performance. All of
2. Design

The administrative tasks and maintenance are taken care of by the service provider so that all the user or application owner needs to do is use the database. The DBaaS was used for as a cloud DB which was connected to the PaaS where the application was deployed.

2.1.7 Containerization

Containerization is a lightweight alternative to full machine virtualization that involves encapsulating an application in a container with its own operating environment. This provides many of the benefits of loading an application onto a virtual machine, as the application can be run on any suitable physical machine without any worries about dependencies. In case of CookBook application this technology is used in CI/CD.

2.2 Architecture

The CookBook system consists of a central database and the Java Web Application. The application and the database can run on different servers or share the same, depending on the restaurant needs. Also, the web application and database can be deployed on PaaS or live in local server of the restaurant. Possible deployment model demonstrated on 2.1

The application architecture 2.2 is based on 3-layers architecture. This architecture type has its benefits in the development process. It makes possible to update the technology stack of one layer, without impacting other areas of the application. It provides an ease of maintenance of the code base, managing presentation code and business logic separately, so that a change to business layer, does not impact the presentation layer and vice versa. Also, it allows for different development teams to each work on their own areas of expertise. It gives an ability to scale the application up and out. A separate back-end layer, allows to deploy to a variety of databases instead of being locked into one particular technology.

- Presentation layer - layer responsible for presentation of data to the user
- Business layer - layer responsible for all business logic of the application procedure
- Data layer - layer responsible for data persistence

The layers are isolated using interfaces:

- ControllerInterface - interface between the presentation and business layers. It consists of several concrete Controller interfaces for individual parts of the business logic.
- DAOs - public interfaces that extends CrudRepository of each Entity
2.3 Database

As it was discussed above, the CookBook application is required new database schema.

The table User represents the registered user in the system. It consists of first name, last name, email, username, password and user's permissions. User can be registered as a Cook or Manager.

Menu represents a set of meals planned to be cooked and served at the specific day. Menu contains the id of Manager and Cook, date and state set on the process of Menu creation by Cook and reviewing by Manager.

Menu_comment table represents the comments left by Cook or Manager during the process of Menu creation and reviewing. It contains the text of the comment, date when this comment was posted and the User.

Menu meals are represented in the table Dish which contains the Menu id and quantity of meals to be cooked and served. Every Dish contains Recipe id which is used in the meal.

The Recipe table represents the list of recipes existing in the system which
2. Design

Figure 2.2: Architecture

can be used in Dish creation during menu planning. It contains the name of the recipe, short description, instructions for cooking, category and note if this recipe is used for vegetarian meals.

Every Recipe is consists of Ingredients used in recipe. It is represented by ingr_in_recipe table. This table contains the quantity of used ingredient, recipe id and the used ingredient id.

Ingredient table contains the name of the ingredient and it’s weight units.

In order to be able to create a request for ingredients the table that represents the requested ingredient is needed. Ingr_in_request table contains the id of the requested ingredient, quantity of this ingredient which is necessary to supply and the Request id.

The Request table represents the request which contains the date of cre-
2.3. Database

Figure 2.3: Database Schema

All the ingredients used during planning are stored in the stock. The stock_item table represents the list of ingredients currently available. It contains the ingredient id, expiration date and the quantity of the ingredient with given expiration date.

The relations between the tables will be discussed in the Implementation Details [3.2] section.
2.4 Class model

In this section will be shown examples of classes using class models.

2.4.1 Menu Controller Class

The figure 2.4 shows the Menu Controller class which was decided to be shown as an example as one of the most nontrivial and crucial classes. This class is one of the Controllers in CookBook application architecture.

This class implements the Menu Interface containing all the functionality related to all the Menu processes. Usage of the Data Access Object (DAO) pattern allows to isolate the business layer from the persistence(data) layer.
2.4.2 MainView Class

Another interesting example is a MainView class implemented using Java programming language. This class is one of the Views in CookBook application architecture. It contains the functionality for rendering the view and functionality for data presentation. This class implements the View interface of Vaadin framework and extends the Vaadin’s Vertical Layout. All the interfaces for controlling are provided.

![Figure 2.5: Main View Class](image)

2.5 Sequence model

This section contains the sequence diagrams for two sampled use case scenarios. As examples were chosen Add Recipe and Get Overview scenarios.
2. Design

2.5.1 Add Recipe

The Figure 2.6 is a sequence model for the Add Recipe scenario. Cook wants to Add new Recipe to the system. This scenario starts when the Cook has already logged in to the system and the RecipeView is shown. Cook presses the Add Recipe button and the RecipeForm for new recipe creation is opened. Cook should fill all the information about the recipe he wants to add. It contains the name, short description, instructions, categories and if the recipe is for vegetarian dish. Then the ingredients should be added to this recipe. Cook presses the button Update Ingredients and the new RecipeIngredientsForm is shown which contains the list of all ingredient from which Cook can choose. When the ingredients are chosen and quantity is specified, Cook presses Add
Ingredients button. This action closes the RecipeIngredientsForm, returns to RecipeForm and now ingredients are added to the new recipe. Cook presses the Save recipe button. All the information about new recipe is collected and sent to \texttt{recipeInterface} using \texttt{addRecipe()} function. The \texttt{recipeInterface} asks \texttt{recipeDAO} to save the provided recipe to the Database. When it is done and new recipe is saved to the database, the RecipeForm closes and now Cook returned to RecipeView which renders the grid filled with new data. This is the end of the main Add Recipe scenario. Alternatively, Cook can decide not to Add the recipe. It starts when the RecipeForm is shown. Cook can press the Cancel button which will close the RecipeForm and return to the Recipe View.

### 2.5.2 Get Overview

The Figure[2.7] is a sequence model for the Get Overview scenario. The scenario starts when the user is logged in to system and OverviewView is shown. User is any person who uses the system. It can be Cook or Manager. User chooses the date of the menu on which he wants to get an overview and presses the button Get Overview. The date of the menu is send to the \texttt{menuInterface}. Since there is only one possible submitted or approved menu, it is possible to get actual menu just by date. \texttt{MenuDAO} returns the Menu to \texttt{menuInterface}, where all the data is modeled and sent back to the OverviewView. The same case with the ingredient availability. The function \texttt{getNumOfAvailableIngr} returns the map that contains the amount of each used ingredient at stock. Then, this data is prepared for being showed and the overview grid is rendered containing the planned usage, potential usage, real usage, stock availability and total usage of each used ingredient. Again, there is an alternative scenario when the user wants to get the overview for the period of time. He chooses the period and the date from which he would like to get an overview. Then, all the information about ingredient usages and availability for the chosen period is collected. This data then is prepared to be shown by \texttt{prepareData()} where everything is calculated and prepared. This data is used during the table rendering which is shown to the user.
Figure 2.7: Get Overview Sequence Model
Chapter 3

Implementation

In this section the tools choice, implementation details and testing of the CookBook application are discussed.

3.1 Tools

This section contains the list of tools used during the implementation process.

- **JDK 1.8**
  The JDK is a development environment for building applications, and components using the Java programming language.\[7\] It includes tools useful for developing and testing programs written in the Java programming language and running on the Java platform.

- **Maven**
  Maven is a powerful project management tool that is based on POM (project object model). It is used for project build, dependency and documentation. \[8\]

- **JUnit**
  Junit is widely used testing framework along with Java Programming Language. It is used as automation framework for unit testing.

- **GitLab VCS**
  GitLab Version Control System is used to keep track of the changes during the software development process. \[9\]

- **GitLab CI/CD**
  GitLab Continuous Integration/Continuous Delivery.\[9\] CI packages and tests software builds and alerts if the recent changes failed any unit tests. CD is the automation that delivers changes to infrastructure. Is used for building and testing every commit to GitLab and also for deployment to test environment.
3. **Implementation**

- **Heroku**
  Heroku provides a platform as a service (PaaS).\[10\] The benefits and the reason of using PaaS were discussed in the Technologies section 2.1.2

- **ClearDB**
  ClearDB provides a database as a service (DBaaS).\[11\] The benefits and the reason of using DBaaS were discussed in the Technologies section 2.1.2

- **XAMPP**
  XAMPP is a tool that provides the Apache web server, MySQL, phpMyAdmin and control over this software.\[12\] This tool ease the development process.

- **phpMyAdmin**
  phpMyAdmin is used as a database administration tool.\[13\]

- **MySQLWorkbench**
  MySQLWorkbench is a full-featured administration tool for database.\[14\] This tool is used for connection and manipulation with ClearDB.

- **IntelliJ IDEA**
  IntelliJ IDEA is Java integrated development environment (IDE) for developing computer software. This was used as a main IDE.\[15\]

- **Docker**
  Docker is a tool designed to make it easier to create, deploy, and run applications by using containers.\[16\] The benefits of using containers and the reasons were discussed in the Technologies section 2.1.2

### 3.2 Implementation Details

In this section will be discussed several points faced during the implementation process.

#### 3.2.1 Maven

One of the challenges faced during the implantation, was the introduction of the build script. For this task was chosen Maven. Instead of downloading and placing the jars manually in to the project, maven can automate this process defining the dependencies in the pom.xml(Projeto object model). Maven requires internet to download the jars, which is a one time process. The example below shows the several dependencies for the Spring Boot framework.

```xml
<dependencies>
  <dependency>
  </dependency>
</dependencies>
```
3.2. Implementation Details

```xml
<groupId>org.springframework.boot</groupId>
<artifactId>spring-boot-starter-data-jpa</artifactId>
</dependency>
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-actuator</artifactId>
</dependency>
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-jdbc</artifactId>
</dependency>
```

Another feature of maven is its direct purpose. Execution of maven build command also runs the packaging phase. The packaging parameters can also be defined in the pom.xml as it shown below. It creates the JAR artifact of the application in the target directory of the project.

```xml
<groupId>eu.cz.cvut.fit.cb</groupId>
<artifactId>CookBook</artifactId>
<version>0.0.5-SNAPSHOT</version>
<packaging>jar</packaging>
```

All those features made the establishment of the CI/CD process simple. Maven does all the job needed. The code below shows the example of CI/CD of the maven project. Having the execution environment prepared and ready the stages simply contain couple of lines of codes. The build is run with mvn compile and tests run with mvn test.

```yaml
build:
  stage: build
  script:
    - mvn compile

test:
  stage: test
  script:
    - mvn dependency::tree
    - mvn clean compile
    - mvn install -DskipTests
    - mvn test
```

3.2.2 Hibernate/JPA

One of the reasons why the Hibernate is discussed is that this was the cause of wrong generated database. One of the key features of the Hibernate and
3. Implementation

JPA are association mappings. It can use one-to-one, many-to-one and many-to-many associations. But also, this is one of the reasons why the Hibernate should not be used. When the database design is not controlled, then it results in strange mapping especially, when the developer is not experienced with association mappings which are often the most difficult thing to implement correctly.

The examples below demonstrate the correct mapping between tables Ingredient and StockItem.

The code sample of Ingredient entity:

```java
@Entity(name = "ingredient")
@Table(name = "ingredient")
public class Ingredient implements Serializable {
    private static final long serialVersionUID = 1L;
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;
    private String name;
    private String units;

    @OneToMany(mappedBy = "ingredient")
    @NotFound(action = NotFoundAction.IGNORE)
    @LazyCollection(LazyCollectionOption.FALSE)
    private List<IngrInRecipe> ingrInRecipes;
}
```

The code sample of IngrInRecipe entity:

```java
@Entity(name = "ingr_in_recipe")
@Table(name = "ingr_in_recipe")
public class IngrInRecipe implements Serializable {
    private static final long serialVersionUID = 1L;
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;
    private Double quantity;

    @ManyToOne
    @NotFound(action = NotFoundAction.IGNORE)
    @JoinColumn(name = "ingredient_id")
    private Ingredient ingredient;
}
```

3.3 Testing

In this section the unit testing and manual user testing of the CookBook application are discussed.
3.3. Testing

3.3.1 Unit Tests

One of the reasons why the initial version of the CookBook application contains this huge number of problems is that the code was not covered with unit tests. With the new functionalities being implemented, the already existed ones were not tested properly to confirm that the changes did not cause any unexpected behaviour. That is why all the main functions of the CookBook applications are now covered with the unit tests. Especially, the controller classes where were the main problems. The unit testing is done with the help of JUnit discussed in 3.1 section. All the tests run with the build of the project so it allows to see if the newly implemented functionalities causes any troubles. This is an example of the unit testing of `getListIngredientsUsed` method, of class `MenuController`:

```java
@Test
public void testGetListIngredientsUsed() {
    System.out.println("getListIngredientsUsed");
    MenuController instance = spy(MenuController.class);
    instance.menuDao = spy(MenuDao.class);
    instance.recipeDao = spy(RecipeDao.class);
    IngrInRecipe ingrInRecipe = new IngrInRecipe();
    Ingredient ingredient = new Ingredient();
    ingredient.setName("ingredient");
    ingrInRecipe.setIngredient(ingredient);
    ArrayList<IngrInRecipe> ingrInRecipes = new ArrayList<>();
    ingrInRecipes.add(ingrInRecipe);
    Recipe recipe = new Recipe();
    recipe.setIngredients(ingrInRecipes);
    Dish dish = new Dish();
    dish.setRecipe(recipe);
    ArrayList<Dish> dishes = new ArrayList<>();
    dishes.add(dish);
    Menu menu = new Menu();
    LocalDate date = LocalDate.now();
    menu.setDate(date);
    menu.setDishes(dishes);
    menu.setComments(new ArrayList<>());
    assertTrue(instance.getListIngredientsUsed(menu)
        .contains(ingredient));
}
```

Since all of the components in the system interact closely this test is already allowed to detect problems related to changes in code.
3. Implementation

3.3.2 User Testing

The application was given to three people with the scenarios for tests. There were several scenarios one of which is a Recipe Creation scenario when the Cook should create the recipe and add ingredients.

Preconditions: The Cook should be logged in to the system and be on the RecipeView.

Scenario:

1. Cook should press the button Add Recipe which will show the pop-up window with the fields for the name and the description of the recipe.

2. Fill the form

3. Press the button Add and the new menu is created, the pop up is closed and RecipeView is shown.

4. Cook have to choose the newly created menu by clicking on it in order to add ingredients and instructions.

5. After that Cook have to add the ingredients to the recipe. To do that he have to press the Update Ingredients button and the new pop up window with the list of ingredients is shown.

6. After choosing the ingredients Cook press the button Update and the ingredients are added to the recipe and pop-up window closes.

7. The usage of every ingredient have to be specified.

8. Clicking on Update button, Cook saves the recipe.

After the completion of this scenario by users they gave a feedback. Based on this feedback it became clear that the Recipe creation process is overloaded with the pop up windows and includes too many actions for the user to complete. This led to the changes in views. The pop-ups were merged and now the new scenario is discussed in section 2.5.1.
Conclusion

The complexity of the cooking process in the restaurants has led to the decision of creating an application that will facilitate this. The system, that was created during this work, is designed to help with the parts of the cooking process such as menu planning, recipe management and also ingredients management. The application is designed to help in the interaction of the cook and manager in the processes of menu planning and ingredients management in order to organize and control the cooking process.

In this thesis the initial version of the CookBook application was analysed. The analysis of the implemented functionalities and problems discovered during this process allowed to make a list of requirements which had to be implemented in order to accomplish the tasks assigned to the application. Moreover, the new domain model of the CookBook application was described. Based on the new domain model and functionalities, the overall picture of the application was completed. This allowed to analyse and compare the CookBook application to the existing solutions and see the weak and strong sides of the application.

The architecture of the new version of the application and the technologies used during the development process were discussed. Moreover, several class models, sequence models and the new database model were also described. This allowed to create the final version of the CookBook application.

The final version of the application completely fulfil the goal of this thesis. The application provides the recipe management, menu management and stock management functionalities. The version for demonstration is deployed and can be found on this address: cookbook-wb.herokuapp.com. The documentation and instructions are also provided.

However, the application can be improved. For instance, the statistical analysis of the previous menus can be performed and the result included in the menu planning process as a recommended amount of portions to be cooked in the given day. This will help Cook to perform more precise planning which will result in less interactions with stock. Also, the information about allergens
should be included in recipes. The other important point is mobility. In case if the restaurants decide to use tablets for the application it is better to optimise or even make a special version of the application for mobile devices.
Bibliography


Bibliography


Appendix A

Acronyms

GUI  Graphical user interface
XML  Extensible markup language
JPA  Java Persistence API
API  Application programming interface
ORM  Object-relational mapping
MVC  Model-View-Controller
PaaS  Platform as a Service
DBaaS  Database as a Service
Appendix B

Contents of enclosed CD

- readme.txt ....................... the file with CD contents description
- jar ........................................ the directory with artefact
- manuals .... the directory with installation instructions and user manual
- javadoc ................... file with source code documentation
- scripts .......................... SQL scripts
- models .......................... enterprise architect models
- src ................................. the directory of source codes
- wbdcm .............................. implementation sources
- thesis ................... the directory of LATEX source codes of the thesis
- text .............................. the thesis text directory
- thesis.pdf ...................... the thesis text in PDF format
- thesis.ps ....................... the thesis text in PS format