RecipesGUI

public void init()

public void start(Stage stage) throws Exception

Scenes:

- homeScene - 1st scene
- newUserScene – Registration
- beginScene – Main menu
- searchScene
- createScene
- [modifyScene → vrs 2]
- recipeScene

Event handlers:

Start: login & registration

- loginButton: homeScene -> beginScene - Togin action, if successful
- registerButton: homeScene -> newUserScene - To registration
- register: newUserScene -> beginScene - Registration action, adds a new user, and if successful, advances to main menu

Main actions

- searchButton: beginScene -> searchScene - To recipe search
- searchByName: searchScene: Recipe search action, search by recipe name
- searchByStuff: searchScene: Recipe search action, search by ingredient
- createButton: beginScene -> createScene - Recipe creation action
- addRecipeDetails: recipeScene: Adds recipe's basic details, if successful
- addStuff: recipeScene: Adds recipe’s ingredients, if successful
- addPhase: recipeScene: Adds recipe’s guideline, if successful
- addRecipe: recipeScene: Adds the whole recipe to recipe book (eventually to database), if successful
- listButton: beginScene: Lists all recipes
- [modifyButton → vrs 2]
- modify: recipeScene: No actual modifying function yet, merely informing the user how to proceed, if wants to modify a recipe in this version
- remove: recipeScene: Deleting the selected recipe, if successful

Clear fields

- clearRecipeDetails: recipeScene: Empties the input fields of recipe details
- newStuff: recipeScene: Empties the input fields of recipe ingredients

- newPhase: recipeScene: Empties the input fields of recipe instruction
- newRecipeButton: recipeScene: Empties all the input fields of the recipe and allows to start creating a new recipe

Back to previous

- backReg: newUserScene -> homeScene
- backRecipe: recipeScene -> searchScene
- toStartSearch: searchScene -> beginScene
- toStartRecipe: recipeScene -> beginScene
- toStartCreate: createScene -> beginScene

Closing application in different scenes

- endHome: in homeScene
- endBegin: in beginScene
- endReg: in newUserScene
- endSearch: in searchScene
- endRecipe: in recipeScene
- endCreate: in createScene

private void setRecipeView(Stage stage)

recipeView: Creates the view of all recipe's information: basic details, ingredients and preparation guidelines

private void recipesToList(Stage stage, Label info, ListView<String> view)

beginScene & searchScene: Creates the name list of all recipes to certain views

public void stop()

public static void main(String[] args)

RecipesDB

public String getDBPath()

Returns the name of the path of the database in use.

public boolean createRecipeDB()

Creates the database for recipes with the following tables: Recipes, Stuff, Guidance .

public Map getAllRecipes()

Gets all the recipes from the database.

public boolean addRecipe(Recipe newRecipe, Map ingredients, List instructions)

Enters the recipe details to recipe database, as well as ingredients with measures and guidelines to recipe object.

private void addStuff(Map stuffList, int recipeID) throws SQLException

private void addGuidance(List guidance, int recipeID) throws SQLException

public Recipe searchRecipebyName(String recipeName)

Gets the recipe from the database, when searching by the name of the recipe.

public Map<String, Recipe> searchRecipebyStuff(String stuff)

Gets the recipe from the database, when searching by the ingredient of the recipe.

public boolean deleteRecipe(Recipe removableRecipe)

Deletes a selected recipe from the database, as well the related ingredients and instructions.

private Recipe getRecipe(String recipeName) throws SQLException

private Map<String, Recipe> getByIngredient(String ingredient) throws SQLException

private Map<String, Recipe> getSelectedRecipes(List<Integer> ids)

private void setRecipeDetails()

private void setIngredients()

private void setInstructions()

private boolean getConnected()

private boolean handleError(SQLException s)

UsersInterface

String getDBPath()

Returns the bath of the user database.

boolean createUsersDB()

Creates the user database (if not already created), either the actual or the "fake" database for testing.

boolean addUser(User user) throws SQLException

Adds a user object to user database.

User searchUser(String username) throws SQLException

Searches a user object from the user database by the username.

UsersDB

public String getDBPath()

Gets the path of user database as String.

public boolean createUsersDB()

Creates the database of users with one table: Users.

public boolean addUser(User newUser)

Adds a new user to user database.

public User searchUser(String userName)

Searches a user from the user database with the given username.

private User getUser(String userName) throws SQLException

Main

public static void main(String[] args)

User

public String getFirstname()

Getter, returns the first name of the user.

public String getLastname()

Getter, returns the last name of the user.

public String getEmail()

Getter, returns the email address of the user.

public String getUsername()

Getter, returns the username of the user.

public String getPassword()

Getter, returns the password of the user.

public String toString()

The textual formatting of the User object including first name, lastname, email address, username and password.

Validation

public Info<User, String> validate(String username, String password) throws SQLException

Takes care of checking the right username and password of the user, who tries to log into the system.

public boolean newValidate(User newUser)

Checks, that every input field is filled, when a new user registers to the system.

Info → VALMIS

public void setUser(K user)

Setter, sets an User object.

public void setText(V text)

Setter, sets a String object with information text.

public K getUser()

Getter, gets an User object.

public V getText()

Getter, gets a String field, containing information text.

Recipe

public String getRecipeName()

Getter, returns the name of the recipe.

public int getPortionAmount()

Getter, returns how many portions you will make by following this recipe.

public String getCategory()

Getter, returns the recipe category this recipe belongs to.

public int getId()

Getter, returns the recipe's identifying id number.

public Map getIngredientsAndAmounts()

Getter, returns recipe's ingredients with amounts.

public List getIngredients()

Getter, returns recipe's ingredients.

public List getAmounts()

Getter, returns recipe's ingredient amounts.

public List getInstructions()

Getter, returns recipe's instructions.

public void setID(int recipeID)

Setter, sets the recipes id received from the recipe database.

public void setIngredient(String stuff, String amount)

Setter, sets a pair of ingredient name and amount for the recipe

public void setIngredient(Map newIngredients)

Setter, sets a collection of ingredients for the recipe.

public void setInstruction(String text)

Setter, sets a row of instruction to the recipe adding it on the list.

public void setInstruction(List instructions)

Setter, sets a collection of instructions for the recipe.

public String toString()

The textual formatting of the Recipe object including its category, name and portion amount.

RecipeBook

private void setCategories()

public boolean addRecipe(String name, Recipe newRecipe)

Adds a recipe to the recipe book.

public Map getAllRecipes()

Gets all the recipes added to the book as Map.

public List getCategories()

Gets all recipe categories as List.

Test classes