

Software Requirements Specification

For

Cake Online Shop

Version 1.4 approved

Prepared by

Student: Buca-Ghebaura Elizabetha Alexandrina

Group: CR 4.1 A, S1

16.11.2022

Table of Contents

Table of Contents	ii
Revision History	0
1. Introduction	1
1.1 Purpose	1
1.2 Document Conventions	1
1.3 Intended Audience and Reading Suggestions	2
1.4 Product Scope	2
1.5 References	3
2. Overall Description	3
2.1 Product Perspective	4
2.2 Product Functions	4
2.3 System Features	4
2.3.1 Create Account	4
2.3.2 Log in	5
2.3.3 Log Out	5
2.3.4 Admin add cakes/categories	6
2.3.5 Admin delete cakes/categories	6
2.3.6 Admin update/edit cakes/categories	7
2.3.7 Search cake functionality	7
2.3.8 Buy cake/Checkout functionality – for Order	8
2.3.9 Contact functionality	8
2.3.10 Profile	9
2.3.11 Shopping Cart	9
2.4 User Classes and Characteristics	9
2.5 Operating Environment	10
2.6 Design and Implementation Constraints	10
2.7 Assumptions and Dependencies	10
3. External Interface Requirements	11
3.1 User Interfaces	11
3.2 Hardware Interfaces	17
3.3 Software Interfaces	17
3.4 Communications Interfaces	19
4. Other Non-functional Requirements	19
4.1 Performance Requirements	19
4.2 Safety Requirements	19
4.3 Security Requirements	20
4.4 Software Quality Attributes	20

Revision History

Name	Date	Reason for Changes	Version
Buca Elizabetha	16.11.2022	Document formatting and Introduction Added Purpose, Document Conventions, Intended Audience and Reading Suggestions	1.0
Buca Elizabetha	18.11.2022	Added Product Scope Added References , Product Perspective, Product Functions, Operating Environment	1.1
Buca Elizabetha	21.11.2022	External Interface Requirements	1.2
Buca Elizabetha	21.11.2022	Added System Features	1.3
Buca Elizabetha	11.12.2022	Added Other Nonfunctional Requirements and finishing the document	1.4

1. Introduction

1.1 Purpose

The application for the project will be an online store selling products such as cakes, tarts, cookies, cupcakes etc. The store is intended for all people who want to order products according to their preferences.

The site is implemented using ASP.Net Core 6 MVC, Entity Framework Core, Identity, and for the maintenance part of the progress, Github.

1.2 Document Conventions

No.	Abbreviation	Definition of abbreviation
1	HTML	Hypertext Markup Language
2	HTTP	Hypertext Transfer Protocol
4	CSS	Cascading Style Sheets
5	SRS	Software Requirement Specification
6	MySQL	Structured Query Language
7	ASP.NET	Active Server Pages Network Enabled Technologies
8	DB	Database

Fonts:

- Body: Cambria, size 11
- Title: Cambria, size 18
- Subtitle: Cambria, size 14

1.3 Intended Audience and Reading Suggestions

The document is intended to serve several groups of audiences.

The users can be of two types: readers/members or the administrator of the online shop. The administrator should know the product functions (2.2) and the operating environment (2.5) because he can make some modifications within the application that a user (authenticated or not) cannot.

1.4 Product Scope

The purpose of the document is to develop a web application for the marketing of a cake shop, intended for all people who want to order such products.

The application must be easy to use and very efficient, because it is very important for customers to find the products more easily (for example by searching the name of the product they want).

The application will provide the following benefits:

- collecting data from the products
- generating reports
- providing information via a web portal
- the interior of the system
- // users can view reviews of products but also give reviews – not implemented (I will try)

1.5 References

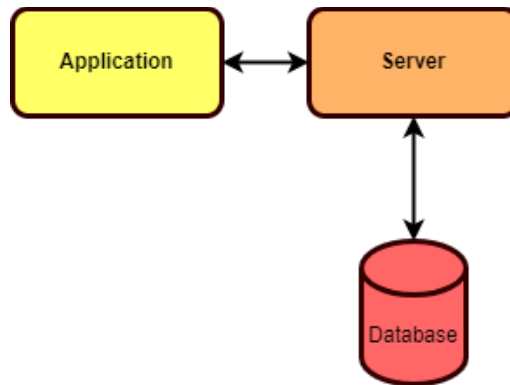
No	Title	Link
1	ASP .NET CORE MVC	https://docs.microsoft.com/en-us/aspnet/core/tutorials
2	Stack overflow	https://stackoverflow.com/
3	YouTube	https://www.youtube.com/
4	Diagrams	https://www.diagrams.net/
5	Learn Microsoft	https://learn.microsoft.com/

2. Overall Description

2.1 Product Perspective

This system consists of a single part, a web application, which will be used by administrator, but also by customers. Also, the application will constantly communicate with its database. The application will be accessed using a computer that has at least the minimum usage specifications application.

Such a structure is shown graphically in the following figure:



2.2 Product Functions

- Users can create an account
- Users can search for the desired product after name
- // → Users can buy products
- // → After the total bill is calculated, the user can checkout - I will try
- // → Users can order a personalized cake according to preferences - I will try
- Users can contact the admin through a message
- Users can change their profile information
- Users can delete the account

- Administrator can add/create, delete, or edit/update a product/category from the collection

2.3 System Features

2.3.1 Create Account

2.3.1.1 Description: The user should be able to register through the application using an email address and a password.

2.3.1.2 Stimulus/Response Sequences:

- Users can register by completing some mandatory fields (email, password). If one of these fields is not completed, the account cannot be created and a message of “The Email field is required.” Or “The Email field is required.” will be displayed.
- If an account with that email address already exists, a message pops up and informs the user that the chosen username is taken.
- To register, a “Register” button must be pressed.

2.3.1.3 Functional Requirements:

- 1) A textbox for each mandatory field must be provided. The mandatory fields are email and password. It should be checked if the email address already exists.
- 2) If it exists, a new account cannot be created, so the program will throw an exception.
- 3) It should be checked if the password contains at least one capital letter, one digit and at least 8 more characters.
- 4) If these conditions are not fulfilled, a new account cannot be created, so the program will throw an exception.
- 5) It should be checked if all the mandatory fields are filled.
- 6) If they are not, an exception is thrown and the account cannot be created.
- 7) When the “Register” button will be pressed, the information is read from the interface.
- 8) The DB will be automatically updated.
- 9) If everything is alright, the application will send the user to the main page.
- 10) The new account which was created will be saved into the database.

2.3.2 Log in

2.3.2.1 Description: The user should be able to login in the application using an email and a password.

2.3.2.2 Stimulus/Response Sequences:

- Users can login by introducing the email and password. If one of these fields is not completed, the user cannot login and a message of “Not all the fields were completed” will be displayed.
- After introducing the email and password, the “Log in” button must be pressed.
- Admins will login like simple users introducing their email address and password. Their specific rights will be available only if that account is marked as “Admin right” in the database.

2.3.2.3 Functional Requirements:

- 11) A textbox for each's mandatory field must be provided. The mandatory fields are: email address, password.
- 12) It must be checked if both email address and password are filled in.
- 13) If one of them (or both) is an empty string, an exception will be thrown.
- 14) It must be checked if the email address and password introduced correspond to the same user.
- 15) If the email address and password do not match for one user, an exception will be thrown.
- 16) If everything is alright, the user must be redirected to the main page.

2.3.3 Log out

2.3.3.1 Description: The user should be able to logout by pressing one button.

2.3.3.2 Stimulus/Response Sequences:

- Users can logout by clicking on the “Log out” button. The user will be redirected to the starting page.

2.3.3.3 Functional Requirements:

- 17) After pressing the “Log out” button, the user will be redirected to the Homepage.

2.3.4 Admin add cake/categories

2.3.4.1 Description: The admin should be able to add cakes/categories

2.3.4.2 Stimulus/Response Sequences:

- Admins will provide information for all mandatory fields.
- Admins will save the information by clicking on “Create” button.
- If one of the fields is not completed, an error message will be displayed.
- If everything works fine, the cakes/categories will be added

2.3.4.3 Functional Requirements:

- 18) A textbox for each mandatory field should be provided.
- 19) Clicking the “Add” button should add the cakes/categories in the database if everything works fine.
- 20) All the available fields should respect the pattern allocated.
- 21) If at least one field doesn't respect the pattern, the program will throw an exception.
- 22) If one textbox is empty, an exception is thrown.

2.3.5 Admin delete cakes/categories

2.3.5.1 Description: The admin should be able to delete cakes/categories

2.3.5.2 Stimulus/Response Sequences:

- Admins will be able to delete some cakes/categories by pressing the “Delete” button.
- If everything works fine, the cakes/categories will be deleted.

2.3.5.3 Functional Requirements:

- 23) A “Delete” button should be provided for each cakes/categories.
- 24) Clicking the “Delete” button should delete the cake/category and update the database
- 25) A message should be displayed that the cake/category has been deleted.

2.3.6 Admin update/edit cakes/categories

2.3.6.1 Description: The admin user should be able to update/edit cakes/categories.

2.3.6.2 Stimulus/Response Sequences:

- Admins will be able to update/edit some information about cakes/categories by pressing the “Update/Edit” button for a specific cake/category.
- Every cake/category will have a button called “Update/Edit”.

2.3.6.3 Functional Requirements:

- 26) A textbox for each field should be provided with the current information in them.
- 27) Every textbox can be selected and the information can be updated.
- 28) If at least one field doesn't respect the pattern, cake/category cannot be updated, so the program will throw an exception.
- 29) If one textbox is empty, an exception is thrown.
- 30) Clicking the “Update/Edit” button should update the cake/category in database.
- 31) A message should be displayed that the cake/category has been updated.

2.3.7 Search functionality

2.3.7.1 Description: The user should be able to search a cake after name.

2.3.7.2 Stimulus/Response Sequences:

- The user should be able to search a cake by its name in the Search bar. Clicking the “Search” button should display the available cakes.

2.3.7.3 Functional Requirements:

- 32) A textbox for each search criteria should be provided
- 33) If the “Search” button is pressed, a checking for the search option should be done in the database
- 34) If a match is found, the cakes with that name should be displayed in the application.
- 35) If no match is found, a suggestive message should be displayed.

2.3.8 Buy cake/Checkout functionality – For Order

2.3.8.1 Description: The user should be able to buy 1 or more categories of cakes.

2.3.8.2 Stimulus/Response Sequences:

- The user should be able to buy 1 or more categories of cakes.

2.3.8.3 Functional Requirements:

- 36) A textbox for each mandatory field should be provided. The mandatory fields are: country, city, address and zip code.
- 37) If at least one textbox is left empty, the program will throw an exception.
- 38) If everything is completed, the “Checkout” button should be pressed.
- 39) More two mandatory fields will pop out for modality of payment
- 40) If at least one textbox is left empty, the program will throw an exception.
- 41) After completing the information, the “Buy” button should be pressed.
- 42) The DB will be updated.

2.3.9 Contact functionality

2.3.9.1 Description: The user should be able to contact the administrator.

2.3.9.2 Stimulus/Response Sequences:

- The user will provide some details and will send an email to the administrator.

2.3.9.3 Functional Requirements:

- 43) A textbox for each mandatory field should be provided. The mandatory fields are: email and message.
- 44) A “Submit” button should be implemented for sending the message.
- 45) All the fields will be completed.
- 46) If at least one field is left empty, the program will throw an exception.

2.3.10 Profile

2.3.10.1 Description: The user can access the section “Profile” where it will display information about them and he will be able to change some of the details.

2.3.10.2 Stimulus/Response Sequences:

- After logging in, the users can change some details about them, like: email address, password.

2.3.10.3 Functional Requirements:

- 47) A textbox for each field will show up to input the information that the user wants to change.
- 48) If some changes are made, the button “Save” must be pressed.
- 49) After the changes, the new fields will be updated in the DB.

2.3.11 Shopping Cart

2.3.11.1 Description: An unauthenticated User cannot access the shopping cart. On the other hand, if we are logged in as a user or admin, the shopping cart can be accessed.

2.3.11.2 Stimulus/Response Sequences:

After logging in, the users should see the products they added, along with the price and quantity and a button for Checkout.

2.3.11.3 Functional Requirements:

- 50) A button must be pressed to be able to enter the Shopping Cart and view the products the user want to buy.

2.4 User Classes and Characteristics

The application has 3 classes:

→ *Guests* - customers which have not logged in, have limited accessibility to the website. They can only search for products or view products; if they want to add a product in the shopping cart, they must log in or sign in.

→ *User* - customers which have an account and are logged in, can buy products and view the orders checked out. Also, the user is able to send messages (contact) to the admin.

→ *Admin* - the administrator will be able to access a database of customers. He can manage products, categories, and update/edit certain parts of it. He can also receive the message sent by the User to the Contact section.

2.5 Operating Environment

- The software being developed will be compatible with machines running on all versions of Windows. Website will run on IISExpress Server.
- Development will be done using **Microsoft Visual Community, MySQL**.
- The website will be written using **HTML, CSS, JavaScript, bootstrap** and **C#**.
- Google Chrome will be the main browser used to test the website.
- All databases and functional components will be stored on a personal computer for testing purposes. Databases and servers are handled by **Microsoft Visual Studio Community & MySQL Server**.
- The website will work on almost any machine, so the minimum requirements are trivial.

2.6 Design and Implementation Constraints

- The application will be constrained by the capacity of the database.
- Integrated Development environment: *Microsoft Visual Studio Community*
- Programming language: **C#**
- Design pattern: **MVC** (Model View Controller)
- Programming User Interface: **.NET 6**
- Relational Database Management System: **MS SQL**
- Software Version Control: **GitHub**
- Communication Protocol: **HTTP**

2.7 Assumptions and Dependencies

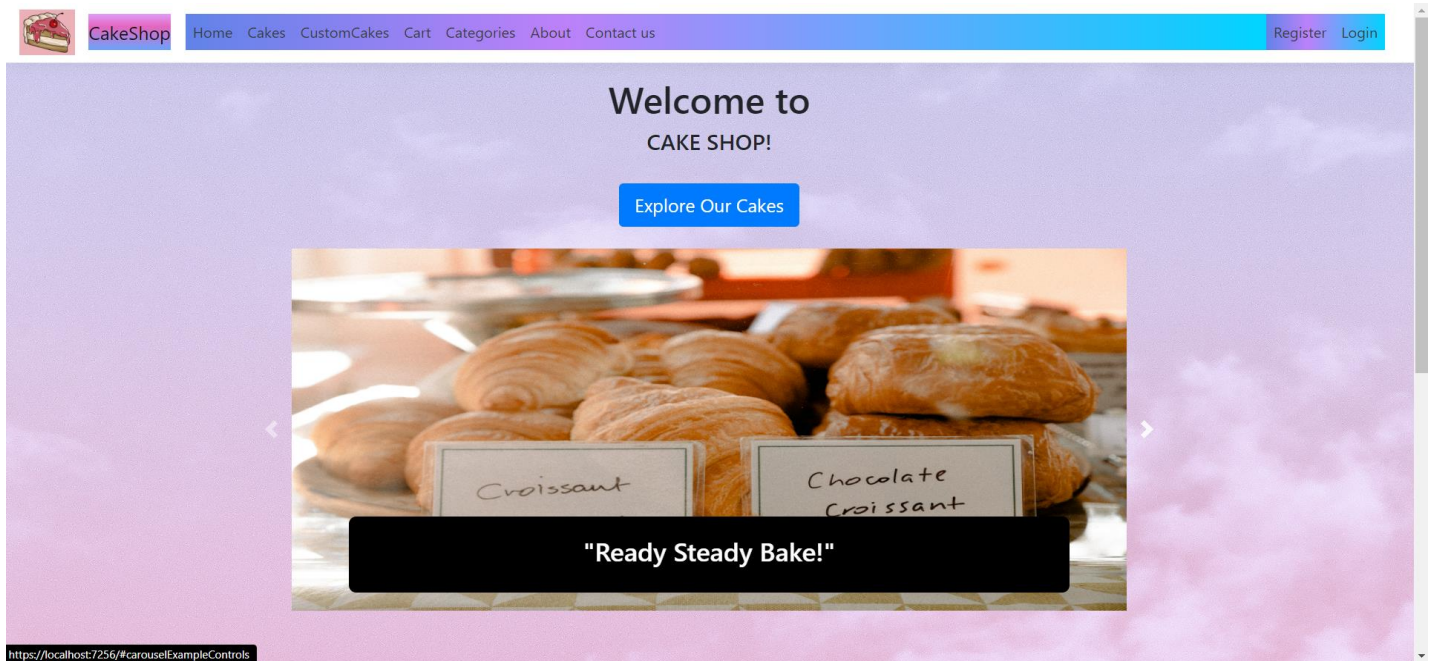
The requirements needed to run the application:

- Computer
- Internet connection
- Web browser Chrome

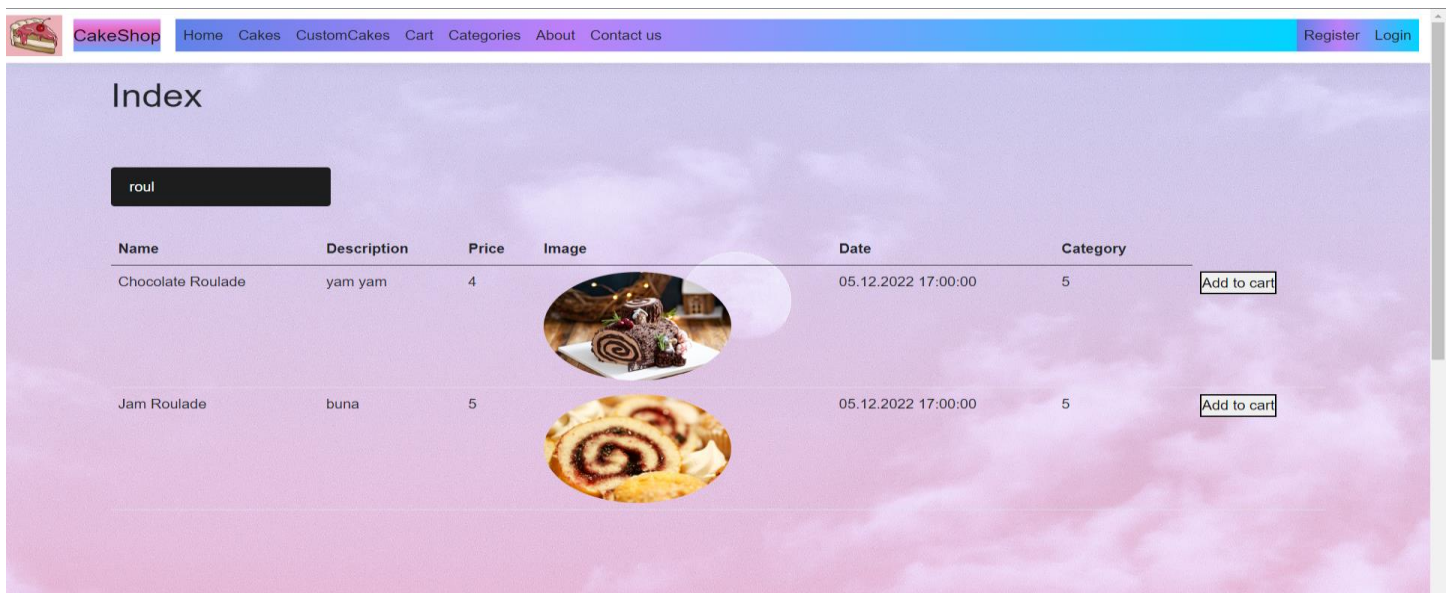
3 External Interface Requirements

3.1 User Interfaces

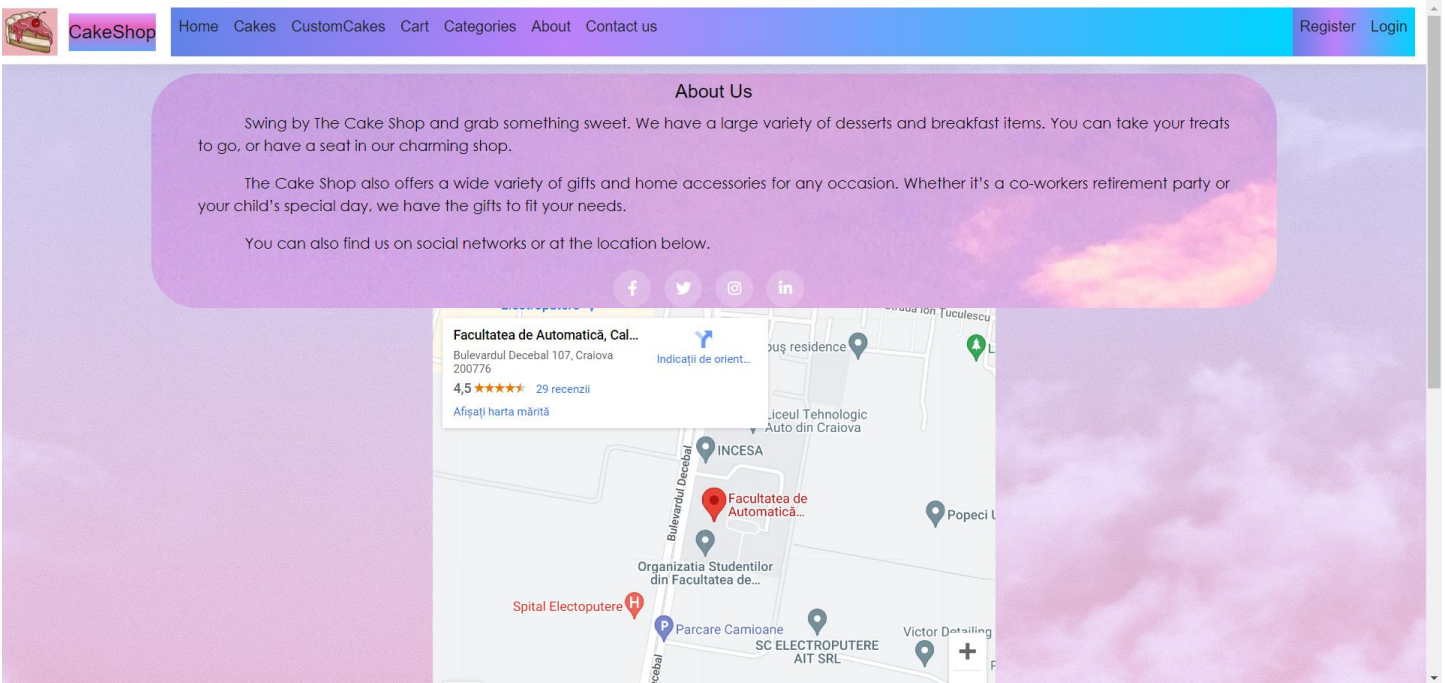
1. "Home" page - which can be viewed by all types of users, including the user who is a guest, i.e. does not have an account in the application (not a member). The Home page contains information about the special offers available.



2. "Search Cake page - this page has a button called suggestive ("Select a Cake") through which will be displayed the available cake about which certain details can be viewed.



3. "Shopping Cart" page - which can be accessed as specified above through the "Home" page or directly from the navbar. If we are not logged in, we will be sent to create an account or to connect. If we are already logged in, then we will be sent to a page where all the products/cakes added to the shopping cart will be displayed, along with the total price and a "Send order" button.
4. "About" page - where you can find details about the shop and useful links for Facebook, Instagram pages.



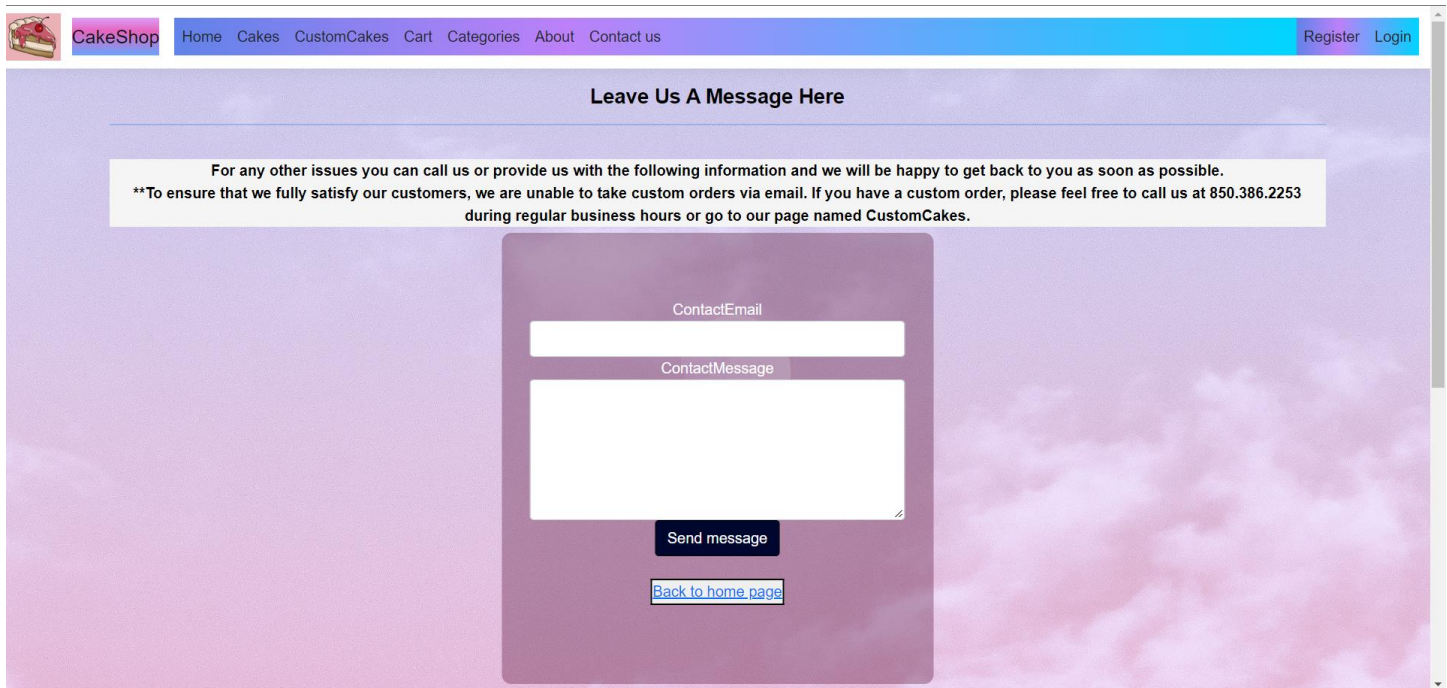
For the integration of the google maps map, I have made a div class called "*maprouter*" through which, with the help of an iframe, I would insert the source of the page from google maps along with some styles for position and size.

```

<div class="mapouter">
  <div class="gmap_canvas">
    <iframe src="https://www.google.com/maps/embed?pb=!1m18!1m12!1m3!1d2855.0574171509325!2d23.832891415272965!3d44.30877011780959!2m3!1f0!2f0!3f0!3m2!1i1024!2i1024!3m2!1s1024!2s1024!3s1024!4s1024!5s1024!6s1024!7s1024!8s1024!9s1024!10s1024!11s1024!12s1024!13s1024!14s1024!15s1024!16s1024!17s1024!18s1024!19s1024!20s1024!21s1024!22s1024!23s1024!24s1024!25s1024!26s1024!27s1024!28s1024!29s1024!30s1024!31s1024!32s1024!33s1024!34s1024!35s1024!36s1024!37s1024!38s1024!39s1024!40s1024!41s1024!42s1024!43s1024!44s1024!45s1024!46s1024!47s1024!48s1024!49s1024!50s1024!51s1024!52s1024!53s1024!54s1024!55s1024!56s1024!57s1024!58s1024!59s1024!60s1024!61s1024!62s1024!63s1024!64s1024!65s1024!66s1024!67s1024!68s1024!69s1024!70s1024!71s1024!72s1024!73s1024!74s1024!75s1024!76s1024!77s1024!78s1024!79s1024!80s1024!81s1024!82s1024!83s1024!84s1024!85s1024!86s1024!87s1024!88s1024!89s1024!90s1024!91s1024!92s1024!93s1024!94s1024!95s1024!96s1024!97s1024!98s1024!99s1024!100s1024!101s1024!102s1024!103s1024!104s1024!105s1024!106s1024!107s1024!108s1024!109s1024!110s1024!111s1024!112s1024!113s1024!114s1024!115s1024!116s1024!117s1024!118s1024!119s1024!120s1024!121s1024!122s1024!123s1024!124s1024!125s1024!126s1024!127s1024!128s1024!129s1024!130s1024!131s1024!132s1024!133s1024!134s1024!135s1024!136s1024!137s1024!138s1024!139s1024!140s1024!141s1024!142s1024!143s1024!144s1024!145s1024!146s1024!147s1024!148s1024!149s1024!150s1024!151s1024!152s1024!153s1024!154s1024!155s1024!156s1024!157s1024!158s1024!159s1024!160s1024!161s1024!162s1024!163s1024!164s1024!165s1024!166s1024!167s1024!168s1024!169s1024!170s1024!171s1024!172s1024!173s1024!174s1024!175s1024!176s1024!177s1024!178s1024!179s1024!180s1024!181s1024!182s1024!183s1024!184s1024!185s1024!186s1024!187s1024!188s1024!189s1024!190s1024!191s1024!192s1024!193s1024!194s1024!195s1024!196s1024!197s1024!198s1024!199s1024!200s1024!201s1024!202s1024!203s1024!204s1024!205s1024!206s1024!207s1024!208s1024!209s1024!210s1024!211s1024!212s1024!213s1024!214s1024!215s1024!216s1024!217s1024!218s1024!219s1024!220s1024!221s1024!222s1024!223s1024!224s1024!225s1024!226s1024!227s1024!228s1024!229s1024!230s1024!231s1024!232s1024!233s1024!234s1024!235s1024!236s1024!237s1024!238s1024!239s1024!240s1024!241s1024!242s1024!243s1024!244s1024!245s1024!246s1024!247s1024!248s1024!249s1024!250s1024!251s1024!252s1024!253s1024!254s1024!255s1024!256s1024!257s1024!258s1024!259s1024!260s1024!261s1024!262s1024!263s1024!264s1024!265s1024!266s1024!267s1024!268s1024!269s1024!270s1024!271s1024!272s1024!273s1024!274s1024!275s1024!276s1024!277s1024!278s1024!279s1024!280s1024!281s1024!282s1024!283s1024!284s1024!285s1024!286s1024!287s1024!288s1024!289s1024!290s1024!291s1024!292s1024!293s1024!294s1024!295s1024!296s1024!297s1024!298s1024!299s1024!300s1024!301s1024!302s1024!303s1024!304s1024!305s1024!306s1024!307s1024!308s1024!309s1024!310s1024!311s1024!312s1024!313s1024!314s1024!315s1024!316s1024!317s1024!318s1024!319s1024!320s1024!321s1024!322s1024!323s1024!324s1024!325s1024!326s1024!327s1024!328s1024!329s1024!330s1024!331s1024!332s1024!333s1024!334s1024!335s1024!336s1024!337s1024!338s1024!339s1024!340s1024!341s1024!342s1024!343s1024!344s1024!345s1024!346s1024!347s1024!348s1024!349s1024!350s1024!351s1024!352s1024!353s1024!354s1024!355s1024!356s1024!357s1024!358s1024!359s1024!360s1024!361s1024!362s1024!363s1024!364s1024!365s1024!366s1024!367s1024!368s1024!369s1024!370s1024!371s1024!372s1024!373s1024!374s1024!375s1024!376s1024!377s1024!378s1024!379s1024!380s1024!381s1024!382s1024!383s1024!384s1024!385s1024!386s1024!387s1024!388s1024!389s1024!390s1024!391s1024!392s1024!393s1024!394s1024!395s1024!396s1024!397s1024!398s1024!399s1024!400s1024!401s1024!402s1024!403s1024!404s1024!405s1024!406s1024!407s1024!408s1024!409s1024!410s1024!411s1024!412s1024!413s1024!414s1024!415s1024!416s1024!417s1024!418s1024!419s1024!420s1024!421s1024!422s1024!423s1024!424s1024!425s1024!426s1024!427s1024!428s1024!429s1024!430s1024!431s1024!432s1024!433s1024!434s1024!435s1024!436s1024!437s1024!438s1024!439s1024!440s1024!441s1024!442s1024!443s1024!444s1024!445s1024!446s1024!447s1024!448s1024!449s1024!450s
```


Software Requirements Specification for Cake Shop Page 13

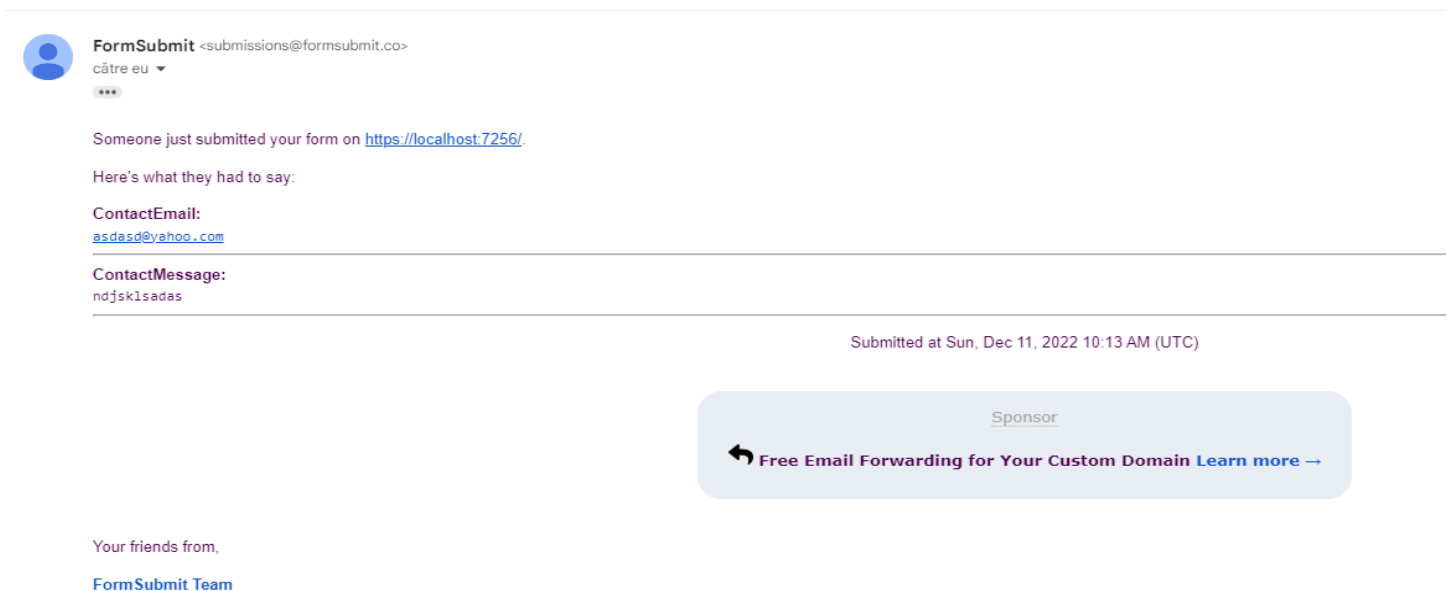
5. "Contact" page - the page through which any type of user can contact us for certain questions/complaints, etc. It is necessary to fill in the 2 fields with suggestive names ("Email" and "Message") in order to get in touch with the admin.



The screenshot shows a web browser window with the 'CakeShop' logo and navigation menu (Home, Cakes, CustomCakes, Cart, Categories, About, Contact us). The 'Contact us' page has a purple header with the text 'Leave Us A Message Here'. Below this is a message box stating: 'For any other issues you can call us or provide us with the following information and we will be happy to get back to you as soon as possible. **To ensure that we fully satisfy our customers, we are unable to take custom orders via email. If you have a custom order, please feel free to call us at 850.386.2253 during regular business hours or go to our page named CustomCakes.' The form contains two input fields: 'ContactEmail' and 'ContactMessage'. Below the fields is a 'Send message' button and a 'Back to home page' link.

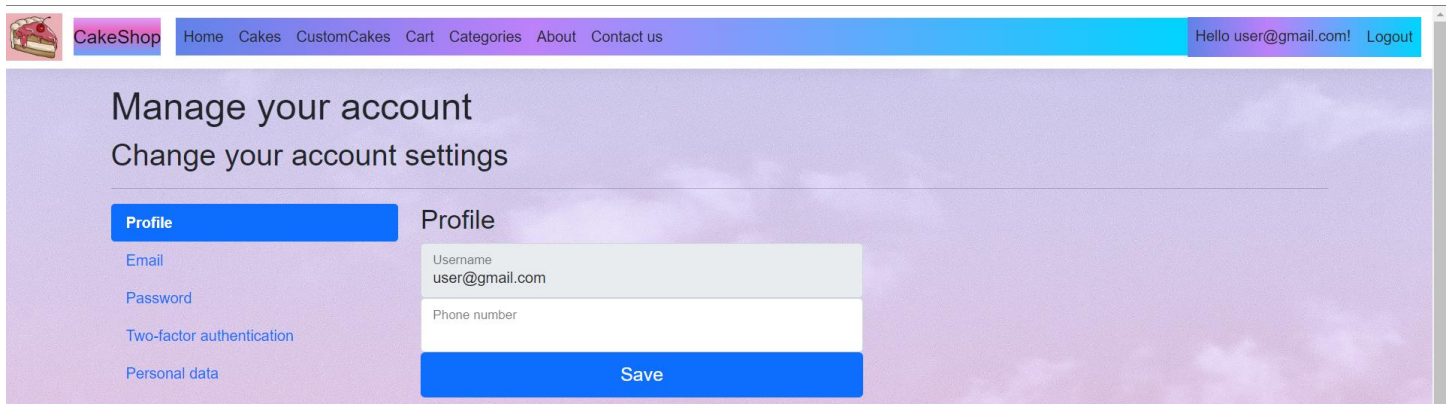
For this page, any User (authenticated, unauthenticated) can send a message to the Admin, entering an email and the message. After pressing the "Send message" button, a pop-up will appear, saying that the message has been sent.

When a person sends a message, the Admin will receive a copy on the email address with the message sent together with the respective user's email.



The screenshot shows an email notification from FormSubmit. The header includes the FormSubmit logo and the email address <submissions@formsubmit.co>. The body of the email states: 'Someone just submitted your form on <https://localhost.7256/>. Here's what they had to say:'. Below this, the 'ContactEmail' is listed as asdasd@yahoo.com and the 'ContactMessage' is 'ndjsklisadas'. The footer indicates the submission time: 'Submitted at Sun, Dec 11, 2022 10:13 AM (UTC)'. At the bottom, there is a 'Sponsor' section with a link to 'Free Email Forwarding for Your Custom Domain' and a 'Learn more' link.

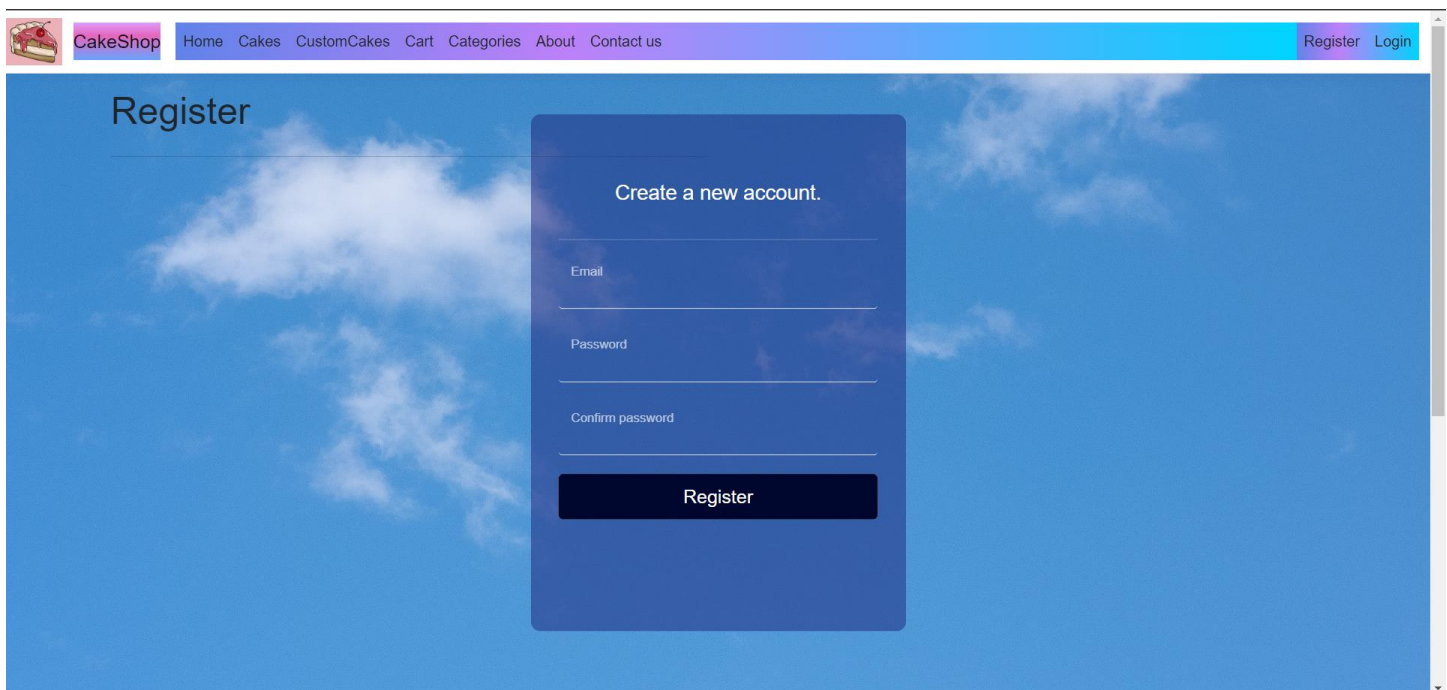
6. "Profile" page - the page available only to members who have created an account on the site. Here you can view your account details and change them at any time.



The screenshot shows the 'Manage your account' page. The header includes the 'CakeShop' logo, navigation links (Home, Cakes, CustomCakes, Cart, Categories, About, Contact us), and a user greeting 'Hello user@gmail.com!' with a 'Logout' link. The main content area has a title 'Manage your account' and a subtitle 'Change your account settings'. On the left, there is a sidebar with links: 'Profile' (highlighted), 'Email', 'Password', 'Two-factor authentication', and 'Personal data'. The 'Profile' section on the right contains a form with fields for 'Username' (pre-filled with 'user@gmail.com') and 'Phone number'. A blue 'Save' button is at the bottom of the form.

7. "Log in/Register" page - where you can log in/out or sign in. Within this page there are 2 options: log in to an existing account and create a new account. For these there is a specific interface. For login there are 3 fields that must be filled in ("email address", "password" and "confirm password") and for registration there are 2 fields that must be filled in ("email address" and "password"). In addition, if a certain account has rights by admin, he will log in as a regular user but will have various benefits in terms of editing information on the site.

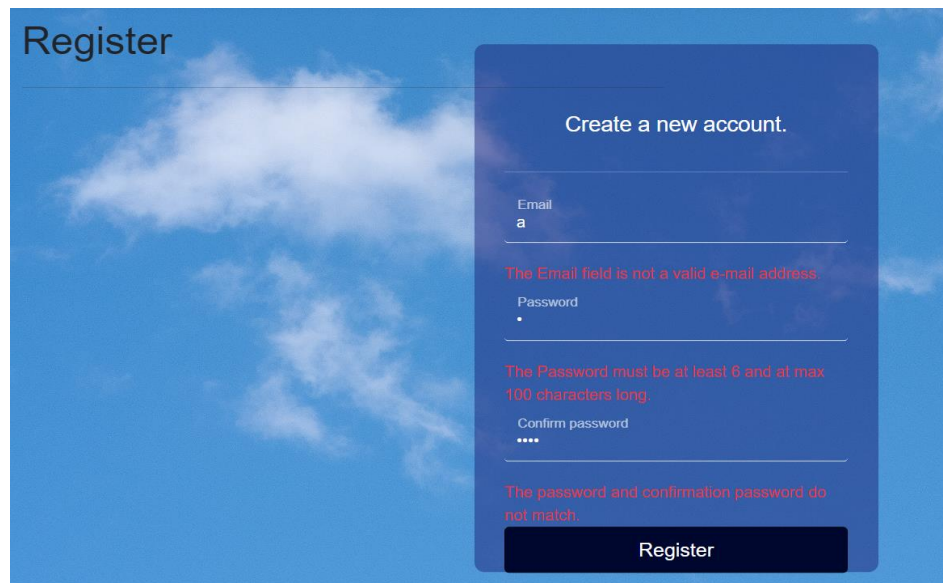
Register



The screenshot shows the 'Register' page. The header is identical to the previous page, but the user greeting is 'Register Login'. The main content area has a title 'Register'. A dark blue modal form is centered on the page with the title 'Create a new account.'. The form contains three input fields: 'Email', 'Password', and 'Confirm password'. A dark blue 'Register' button is at the bottom of the form.

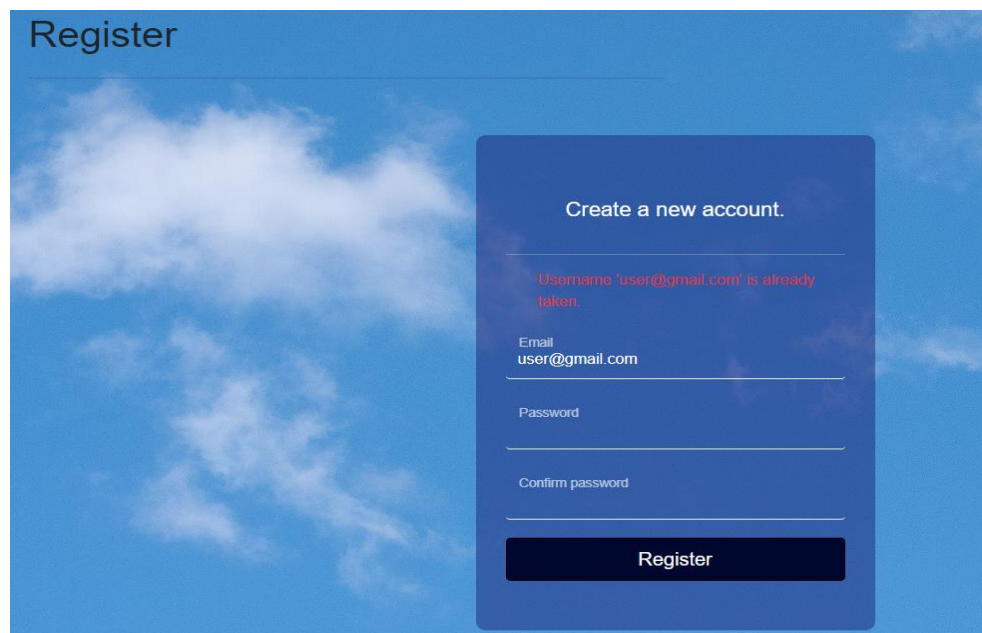
For this page, we have configured some conditions for which a user can create an account:

- Email - if it does not correspond to the way an email is written, then a suggestive message will appear with **"The Email field is not a valid e-mail address."**
- Password - must have at least 6 characters. If you do not respect this, the message **"The Password must be at least 6 and at max 100 characters long."** will appear.
- Confirmed password - must match the password from the previous field. Otherwise, it will be written **"The password and confirmation password do not match."**



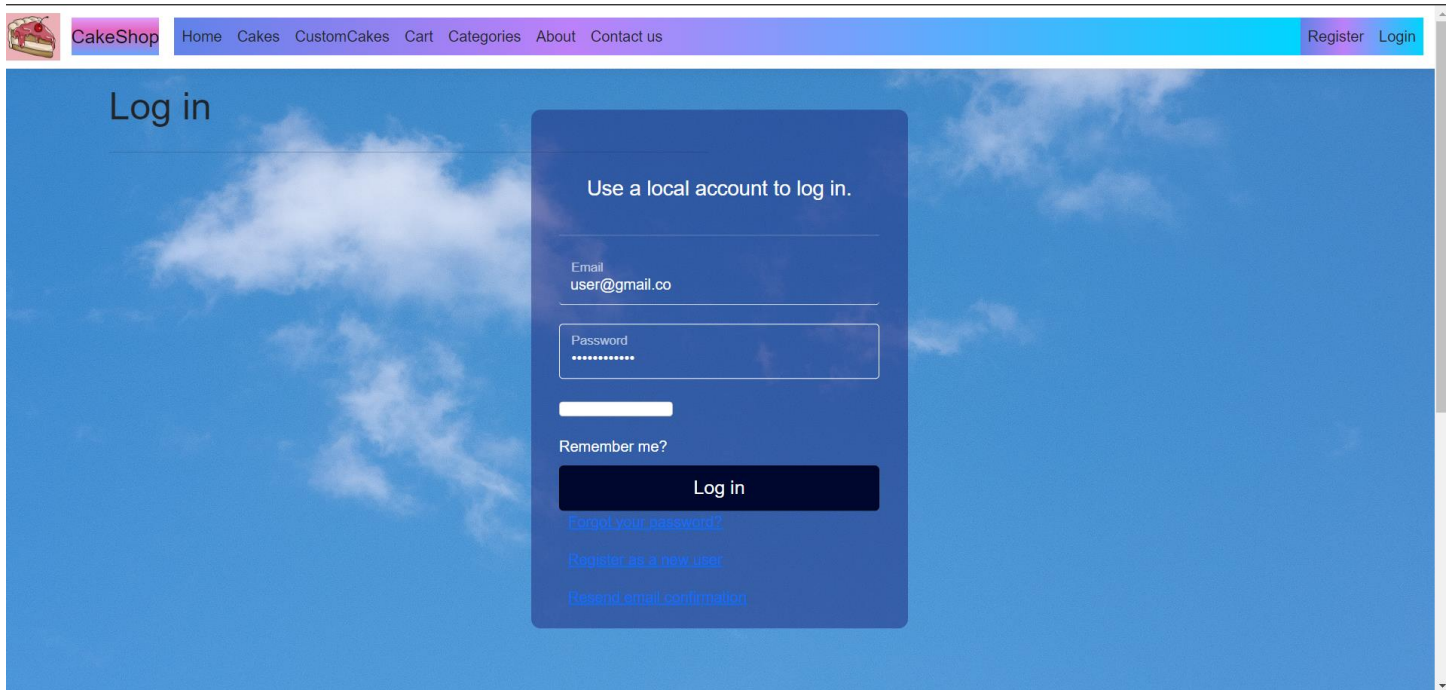
The screenshot shows a 'Register' form titled 'Create a new account.' on a blue background with clouds. The form has three input fields: 'Email' with the value 'a', 'Password' with a single dot, and 'Confirm password' with four dots. Below each field is a red error message: 'The Email field is not a valid e-mail address', 'The Password must be at least 6 and at max 100 characters long.', and 'The password and confirmation password do not match.' respectively. A dark blue 'Register' button is at the bottom.

If the account exists, and we try to create a new account using the same email address, a message will be displayed. Example: **Username 'user@gmail.com' is already taken.**

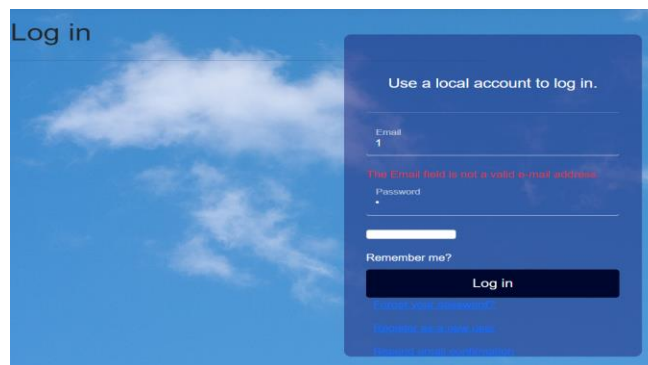


The screenshot shows the same 'Register' form. At the top, a red message states 'Username 'user@gmail.com' is already taken.' Below this, the 'Email' field contains 'user@gmail.com'. The 'Password' and 'Confirm password' fields are empty. The 'Register' button remains at the bottom.

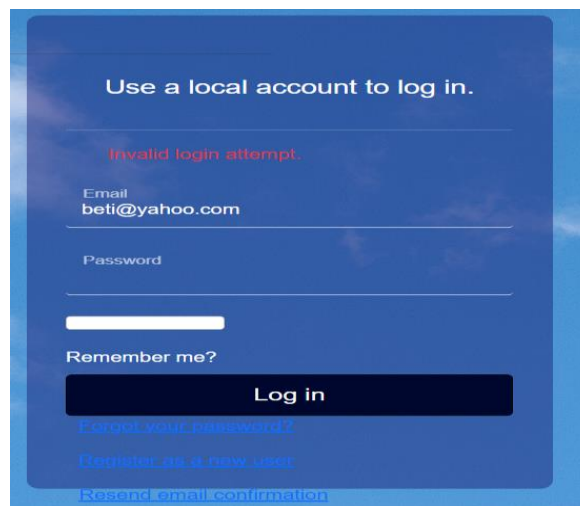
Login



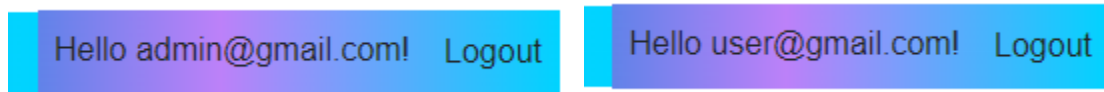
Similar to the Register page, we have 2 fields, Email and Password. In the same way, the email must respect the valid form for an email; otherwise, a message appears.



If we try to log in with an account that does not exist, a message will appear:



If we log in with an existing account, then we will be able to navigate the website pages. A suggestive message will appear on the right side of the screen "hello user@gmail.com" next to the Logout button. Also, a user can enter his profile and edit the data.



I have also integrated the authentication part with Facebook and Google for logging in/registering.



3.2 Hardware Interfaces

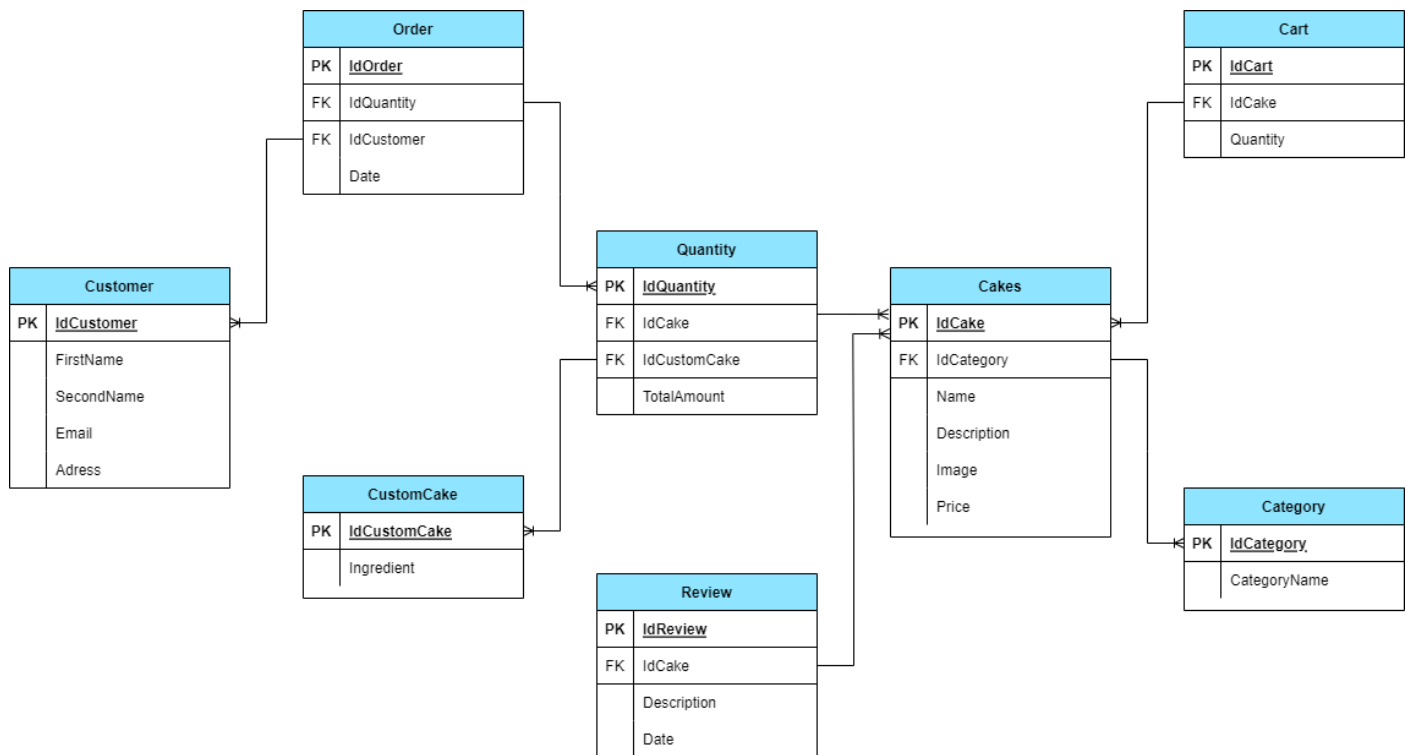
- Hardware Required
 - The software is developed for Windows 7, Windows 8, Windows 10, Android
 - The RAM should be over 2GB
- Supported device types
 - The software is developed for Windows 32-bit or 64-bit

3.3 Software Interfaces

The communication between the database and the application consists of operations concerning both reading and modifying the data. There will be several functions:

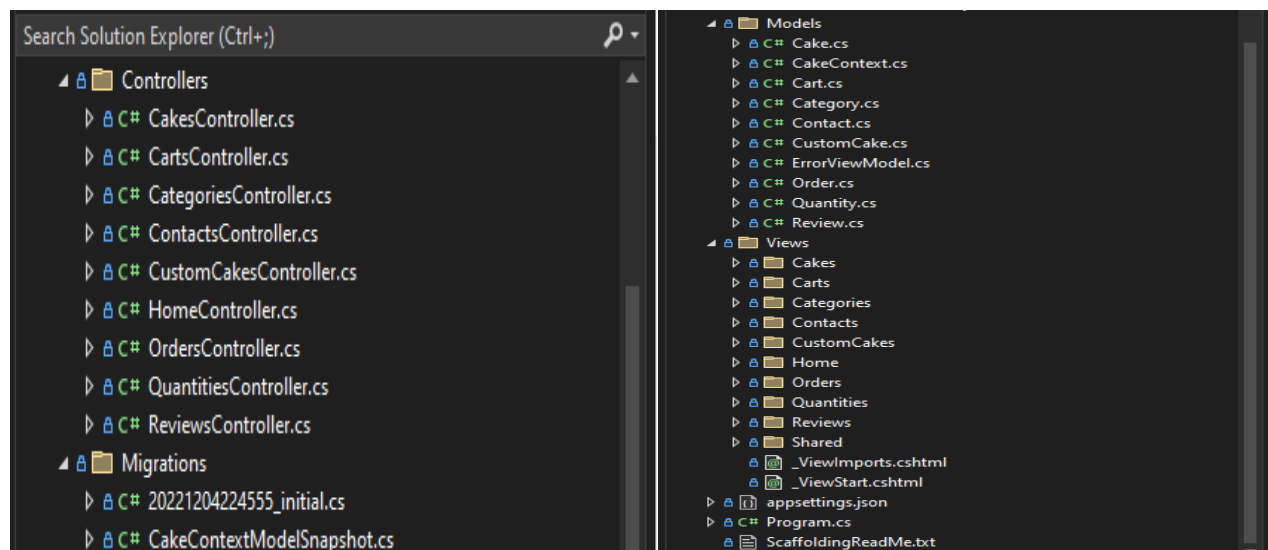
- read from database
- write in database
- delete from database
- add a function that will show the data from the database

The diagram of the database:

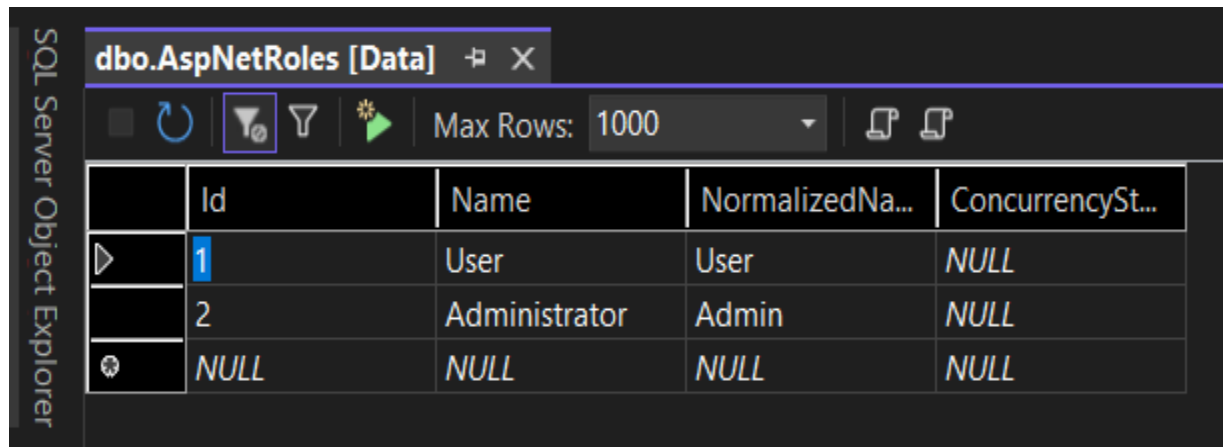


To begin with, I created the database models, I registered all the tables in the Context class, I initialized the database in the Program.cs class and in appsettings.json, I created a migration, and then I updated the database. (according to the CodeFirst strategy)

After creating the actual database, I created the controllers - I selected the database context, the models I wanted to make controllers, then they were generated together with the views.



I then declared 2 types of users, namely User and Admin. There are functionalities that can be used by any type of user (including unauthenticated, authenticated or admin). There will also be functionalities that do not require user authentication, such as viewing the products or the Home page.



The screenshot shows the SQL Server Object Explorer with the 'dbo.AspNetRoles [Data]' table selected. The table has five columns: Id, Name, NormalizedNa..., and ConcurrencySt... (truncated). The data is as follows:

	Id	Name	NormalizedNa...	ConcurrencySt...
▶	1	User	User	NULL
	2	Administrator	Admin	NULL
✱	NULL	NULL	NULL	NULL

3.4 Communications Interfaces

Having that it's a web application, the communication between the client and server is made through HTTP/HTTPS. The web address is set by the IP address and the default port.

4 Other Nonfunctional Requirements

4.1 Performance Requirements

It refers to the fastness of the search results. The user should see an action as fast as possible (in maximum 5 seconds from the click of the button) so he/she can see the desired results.

- The application should load and be usable within 3 seconds
- The application should update the interface on interaction within 5 seconds
- The database should be updated to at most 3 seconds after the "save" button is pressed.
- The system should afford to create a large number of users.

4.2 Safety Requirements

Not applicable.

4.3 Security Requirements

- Users log-in using a unique account and password.
- If a user/admin wants to create an account and the desired email address is used, the user should be asked to choose a different one.
- Normal users can just read information but they cannot edit or modify anything
- Guest users can just read information but they cannot edit or modify anything.
- System will have different types of users and every user has access constraints.
- When creating an account the password should have at least one capital letter, one digit and the size of the password to be at least 8.
- When typing the password no one should be able to see it, so instead of the actual password on the screen will appear “*” for each character or digit.
- The password will be encrypted in the database.

4.4 Software Quality Attributes

Quality code rules:

1. Write only one statement per line, write only one declaration per line
2. Use implicit typing for local variables when the type of the variable is obvious from the right side of the assignment, or when the precise type is not important Use a try-catch statement for most exception handling
3. Use meaningful names for query variables.
4. Each “if” statement should have an else statement.
5. Use KISS (keep it simple, stupid)
6. Use DRY (Don’t repeat yourself) - avoid duplication.
7. Programs should be separated into distinct sections, each addressing a separate concern, or set of information that affects the program.
8. Software entities (classes, modules, functions, etc.) should be open for extension, but closed for modification
9. Subtypes must be substitutable for their base types.
10. A method should have a single purpose.
11. A method should use all provided parameters.
12. A method should either change the state of the object or return a value, not both.

Performance:

13. The web application shall contain smooth animations in drop-down menus, sliders and other related features.
14. The web application shall contain fast execution times for all features.
15. The web application shall manage user login support in a quick and correct manner.