**Dokumentācija.**

**Saturs.**

**1. Problēma.**

**2. Programmatūras prasību specifikācija.**

**3. Programmatūras izstrādes plāns.**

**4. Testēšana.**

**5.** **Lietotāju ceļvedis.**

**6. Programmatūras kods.**

**Problēma.**

Nepārstrādātiem pārtikas produktiem ir ļoti īss derīguma termiņš, un pat ledusskapis nevar tos pasargāt no sabojāšanās. Pārtikas atkritumi bieži rodas tāpēc, ka produkti netiek laikus izlietoti un kļūst nepiemēroti lietošanai, kas var radīt veselības problēmas. Tomēr atcerēties visus ledusskapī esošos produktus un to derīguma termiņus ne vienmēr ir prioritāte. Turklāt, atrodoties veikalā, nav iespējams vienkārši ieskatīties ledusskapī un pārliecināties, vai, piemēram, piens jau nav sabojājies un būtu jāiegādājas jauns.

**Programmatūras prasību specifikācija.**

Mērķauditorija:

* Cilvēki, kuri vēlas ērti un efektīvi pārvaldīt savu ledusskapja saturu digitāli un tam piekļūt jebkurā laikā.
* Lietotāji, kuri vēlas samazināt pārtikas izšķērdēšanu, labāk organizējot produktus un to derīguma termiņus

Programmatūras funkcijas:

* Lietotāju pieteikšanās un reģistrācija.
* Produktu pievienošana
* Produktu dzēšana
* Produktu ar īsu derīguma termiņu izcelšana

Programmatūras skice:

A diagram of a login and sign up

Description automatically generated

Programmēšanas valodas un tehnoloģijas:

* Frontend- HTML, CSS, JavaScript
* Backend- Node.js (Express)
* Datubāze- SQLite3

**Programmatūras izstrādes plāns.**

Izstrādes modelis:

Programmatūras izstrāde izmantoju iteratīvo izstrādes modeli, pakāpeniski pievienojot funkcionalitātes un tās testējot. Izvēlējos šo, jo jau no sākta gala ir iespējams redzēt rezultātus konkrētai funckijai.

Izstrādes plāns:

1. Prasību definēšana
2. Frontend izveide
3. Backend izstrāde un savienošana ar datubāzi
4. Sistēmas uzstādīšana un pielāgošana.

Programmatūras mērķis/ prasības:

Izveidot elektronisku ledusskapja pārvaldības mājaslapu, kas palīdz lietotājiem efektīvāk sekot līdzi pārtikas produktiem un to derīguma termiņiem. Sistēma nodrošinās iespēju:

* Reģistrēties un pieslēgties savam ledusskapim.
* Ievadīt un pārskatāmi apskatīt ledusskapī esošos produktus.
* Dzēst produktus, kas izlietoti vai kuriem beidzies derīguma termiņš.
* Automātiski vizuāli izcelt produktus, kuriem tuvojas derīguma termiņa beigas, lai mazinātu pārtikas atkritumu daudzumu.

Prioritates:

* Lai lietotājs varētu pārvaldīt savu personisko ledusskapi vispirms nepieciešams reģistrēties, pieslēgties.
* Izveidot visas funkcijas ledusskapim.

Datubāze:

A screenshot of a computer

Description automatically generated

Saskarnes un dizains:

A diagram of a login and sign up

Description automatically generated

**Testēšana.**

Visas funkcijas tika veiksmīgi notestētas un konstatētie trūkumi tika novērsti. Programmatūra ir stabila un gatava lietošanai.

Testēšanas veidi:

* Manuālā testēšana
* Integrācijas testēšana

**Lietotāju ceļvedis.**

<https://fridge-5f393e5bb273.herokuapp.com>

Dodieties uz šo mājaslapu.

Reģistrācija:

Ja vēlaties reģistrēties, nospiediet uz teksta “Create an Account” un aizpildiet visus prasītos lauciņus, kad tas izdarīts nospiediet uz pogas “Signup”

Ja esat jau reģistrējies nospiediet uz teksta “Login”

Pieslēgšanās:

Kad esat reģistrējušies ievadiet savu e-pastu un paroli ar kuru piereģistrējāties un spiediet uz pogas “Login”.

Produktu pievienošana:

Lai pievienotu produktu nospiediet uz pogas “ ADD PRODUCT” , aizpildiet visus obligātos laukus ( name, expiry date, quantity, category) un spiediet “SUBMIT”, lai nokļūtu ledusskapja skatā spiediet pogu “BACK”.

Produktu dzēšana:

Lai dzēstu produktu nospiediet uz pogas “DELETE PRODUCT”, tad nospiediet uz krustiņa, kas parādas labajā augšējā stūrī, lai izdzēstu produktu.Lai nokļūtu ledusskapja skatā spiediet pogu “BACK”.

Iziešana:

Lai izietu no profila nospiediet “LOGOUT” pogu, kas atrodas labajā apakšējā stūri ledusskapja skatam.

Derīguma termiņu krāsas:

Derīguma termiņu krāsas ir attiecīgam dienu skaitam līdz termiņa beigām, balts nozīmē, ka ir vēl vairāk par 7 dienām līdz termiņa beigām, oranžs nozīme, ka ir 7 vai mazāk dienas palikušas, sarkans nozīme, ka trīs vai mazāk dienas līdz termiņa beigām.

**Programmatūras kods.**

Index.html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>FRIDGE</title>

<link rel="stylesheet" href="style.css">

</head>

<body>

<!--galvenais-->

<div id="mainView"> <br>

<h1>FRIDGE</h1> <br>

<!--pogas-->

<div class="button-container">

<button onclick="showView('addView')">ADD PRODUCT</button>

<button onclick="showView('editView')">DELETE PRODUCTS</button>

</div> <br>

<!--kategorijas-->

<div id="productsContainer" class="grid-container">

<section id="piens-un-piena-produkti" class="grid-item">

<h2>MILK/ PLANT MILK PRODUCTS</h2>

<div class="category-container"></div>

</section>

<section id="gala" class="grid-item">

<h2>MEAT/ PROTEIN</h2>

<div class="category-container"></div>

</section>

<section id="darzeni" class="grid-item">

<h2>FRUIT AND VEGETABLES</h2>

<div class="category-container"></div>

</section>

<section id="augli" class="grid-item">

<h2>READY TO EAT MEALS</h2>

<div class="category-container"></div>

</section>

<section id="citi" class="grid-item">

<h2>OTHER</h2>

<div class="category-container"></div>

</section>

</div>

<!--logout-->

<a href="/" class="logout-button">LOGOUT</a>

</div>

<!--add-->

<div id="addView" style="display: none;"> <br>

<button onclick="showView('mainView')">⬅ BACK</button>

<% if (error) { %>

<div class="error-message" ><%= error %></div>

<% } %>

<!--forma-->

<form id="productForm" action="/api/fridge-products" method="post"><br>

<div class="input-group">

<label for="productName">&#128292;</label>

<input type="text" id="productName" placeholder="Product name" required>

</div>

<div class="input-group">

<label for="expiryDate">&#128467;</label>

<input type="date" id="expiryDate" required>

</div>

<div class="input-group">

<label for="quantity">🔢</label>

<input type="number" id="quantity" placeholder="Quantity" min="1" required>

</div>

<div class="input-group">

<label for="category">📂</label>

<select id="category" required>

<option value="" disabled selected>Choose category</option>

<option value="piens-un-piena-produkti">MILK/ PLANT MILK PRODUCTS </option>

<option value="gala">MEAT/ PROTEIN</option>

<option value="darzeni">FRUIT AND VEGETABLES</option>

<option value="augli"> READY TO EAT MEALS</option>

<option value="citi">OTHER</option>

</select>

</div>

<div class="input-group">

<label for="icon">&#127822;</label>

<select id="icon">

<option value="🟦">PLACEHOLDER ICON</option>

<option value="🍎">apple</option>

<option value="🍐">pear</option>

<option value="🍊">orange</option>

<option value="🍋">lemoon</option>

<option value="🍌">banana</option>

<option value="🍉">watermelon</option>

<option value="🥑">avocado</option>

<option value="🥖">bread</option>

<option value="🧀">cheese</option>

<option value="🥚">eggs</option>

<option value="🧈">butter</option>

<option value="🥛">milk</option>

<option value="🥩">steak</option>

<option value="🥓">bacon</option>

<option value="🍕">pizza</option>

<option value="🍗">chicken</option>

<option value="🥗">salad</option>

<option value="🍰">cake</option>

<option value="🍣">sushi</option>

</select>

</div>

<div class="input-group">

<label for="notes">📝</label>

<textarea id="notes" placeholder="More info (e.g. for dinner)"></textarea>

</div>

<button type="submit">ADD PRODUCT</button>

</form>

</div>

<!--dzesana-->

<div id="editView" style="display: none;">

<br>

<div class="button-container">

<button onclick="showView('mainView')">⬅ BACK</button>

</div>

<br>

<div id="editProductsContainer" class="grid-container">

<section id="edit-piens-un-piena-produkti" class="grid-item">

<h2>MILK/ PLANT MILK PRODUCTS</h2>

<div class="category-container"></div>

</section>

<section id="edit-gala" class="grid-item">

<h2>MEAT/ PROTEIN</h2>

<div class="category-container"></div>

</section>

<section id="edit-darzeni" class="grid-item">

<h2>FRUIT AND VEGETABLES</h2>

<div class="category-container"></div>

</section>

<section id="edit-augli" class="grid-item">

<h2>READY TO EAT MEALS</h2>

<div class="category-container"></div>

</section>

<section id="edit-citi" class="grid-item">

<h2>OTHER</h2>

<div class="category-container"></div>

</section>

</div>

</div>

<script src="script.js"></script>

</body>

login.html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Login</title>

<link rel="stylesheet" href="style.css">

<script type="text/javascript"src="script.js" defer></script>

</head>

<body>

<div class="wrapper">

<h1>Login</h1>

<% if (error) { %>

<div class="error-message"><%= error %></div>

<% } %>

<form id="form" action="/logindone" method="post">

<div>

<label for="email-input">

<span>@</span>

</label>

<input type="email" name="email" id="email-input" placeholder="Email">

</div>

<div>

<label for="password-input">

<svg xmlns="http://www.w3.org/2000/svg" height="24px" viewBox="0 -960 960 960" width="24px" fill="#e8eaed"><path d="M240-80q-33 0-56.5-23.5T160-160v-400q0-33 23.5-56.5T240-640h40v-80q0-83 58.5-141.5T480-920q83 0 141.5 58.5T680-720v80h40q33 0 56.5 23.5T800-560v400q0 33-23.5 56.5T720-80H240Zm240-200q33 0 56.5-23.5T560-360q0-33-23.5-56.5T480-440q-33 0-56.5 23.5T400-360q0 33 23.5 56.5T480-280ZM360-640h240v-80q0-50-35-85t-85-35q-50 0-85 35t-35 85v80Z"/></svg>

</label>

<input type="password" name="password" id="password-input" placeholder="Password">

</div>

<button type="submit">Login</button>

</form>

<p>New here? <a href="/signup">Create an Account</a></p>

</div>

</body>

</html>

signup.html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Signup</title>

<link rel="stylesheet" href="style.css">

<script type="text/javascript" src="script.js" defer></script>

</head>

<body>

<div class="wrapper">

<h1>Signup</h1>

<% if (error) { %>

<div class="error-message"><%= error %></div>

<% } %>

<form id="form" action="/register" method="post">

<div>

<label for="firstname-input">

<svg xmlns="http://www.w3.org/2000/svg" height="24px" viewBox="0 -960 960 960" width="24px" fill="#e8eaed"><path d="M480-480q-66 0-113-47t-47-113q0-66 47-113t113-47q66 0 113 47t47 113q0 66-47 113t-113 47ZM160-160v-112q0-34 17.5-62.5T224-378q62-31 126-46.5T480-440q66 0 130 15.5T736-378q29 15 46.5 43.5T800-272v112H160Z"/></svg>

</label>

<input type="text" name="firstname" id="firstname-input" placeholder="Firstname">

</div>

<div>

<label for="email-input">

<span>@</span>

</label>

<input type="email" name="email" id="email-input" placeholder="Email">

</div>

<div>

<label for="password-input">

<svg xmlns="http://www.w3.org/2000/svg" height="24px" viewBox="0 -960 960 960" width="24px" fill="#e8eaed"><path d="M240-80q-33 0-56.5-23.5T160-160v-400q0-33 23.5-56.5T240-640h40v-80q0-83 58.5-141.5T480-920q83 0 141.5 58.5T680-720v80h40q33 0 56.5 23.5T800-560v400q0 33-23.5 56.5T720-80H240Zm240-200q33 0 56.5-23.5T560-360q0-33-23.5-56.5T480-440q-33 0-56.5 23.5T400-360q0 33 23.5 56.5T480-280ZM360-640h240v-80q0-50-35-85t-85-35q-50 0-85 35t-35 85v80Z"/></svg>

</label>

<input type="password" name="password" id="password-input" placeholder="Password">

</div>

<div>

<label for="repeat-password-input">

<svg xmlns="http://www.w3.org/2000/svg" height="24px" viewBox="0 -960 960 960" width="24px" fill="#e8eaed"><path d="M240-80q-33 0-56.5-23.5T160-160v-400q0-33 23.5-56.5T240-640h40v-80q0-83 58.5-141.5T480-920q83 0 141.5 58.5T680-720v80h40q33 0 56.5 23.5T800-560v400q0 33-23.5 56.5T720-80H240Zm240-200q33 0 56.5-23.5T560-360q0-33-23.5-56.5T480-440q-33 0-56.5 23.5T400-360q0 33 23.5 56.5T480-280ZM360-640h240v-80q0-50-35-85t-85-35q-50 0-85 35t-35 85v80Z"/></svg>

</label>

<input type="password" name="repeat\_password" id="repeat-password-input" placeholder="Repeat Password">

</div>

<button type="submit">Signup</button>

</form>

<p>Already have an Account? <a href="/">Login</a></p>

</div>

</body>

</html>

script.js

document.addEventListener("DOMContentLoaded", fetchProducts);

//get products

function fetchProducts() {

fetch("/api/fridge-products")

.then((response) => response.json())

.then((products) => {

renderProducts(products);

})

.catch((err) => console.error("ERROR loading products:", err));

}

function renderProducts(products) {

document.querySelectorAll(".category-container").forEach((container) => (container.innerHTML = ""));

//produktu kastes

products.forEach((product) => {

let productWrapper = document.createElement("div");

productWrapper.classList.add("product-box");

productWrapper.setAttribute("data-expiry", product.expiry\_date);

let box = document.createElement("div");

box.classList.add("box");

let iconElement = document.createElement("div");

iconElement.classList.add("icon");

iconElement.innerText = product.icon;

let content = document.createElement("div");

content.classList.add("product-content");

content.innerHTML = `<strong>${product.product\_name}</strong><br>

<strong>${product.expiry\_date}</strong><br>

Quantity: ${product.quantity}<br>

<p class="notes">${product.notes || ""}</p>`;

box.appendChild(iconElement);

box.appendChild(content);

productWrapper.appendChild(box);

//bridinajuma krasa

updateProductColor(productWrapper, product.expiry\_date);

//sadala kategorijas

let categoryContainer = document.querySelector(`#${product.category} .category-container`);

if (categoryContainer) {

categoryContainer.appendChild(productWrapper);

}

//dzesana

let editProductWrapper = document.createElement("div");

editProductWrapper.classList.add("product-box");

let editBox = document.createElement("div");

editBox.classList.add("box");

let deleteButton = document.createElement("button");

deleteButton.textContent = "X";

deleteButton.classList.add("delete-button");

deleteButton.onclick = () => deleteProduct(product.id);

editBox.appendChild(iconElement.cloneNode(true));

editBox.appendChild(content.cloneNode(true));

editBox.appendChild(deleteButton);

editProductWrapper.appendChild(editBox);

let editCategoryContainer = document.querySelector(`#edit-${product.category} .category-container`);

if (editCategoryContainer) {

editCategoryContainer.appendChild(editProductWrapper);

}

});

}

//pievienosana

document.getElementById("productForm").addEventListener("submit", function (event) {

event.preventDefault();

let productName = document.getElementById("productName").value.trim();

let expiryDate = document.getElementById("expiryDate").value;

let quantity = document.getElementById("quantity").value;

let category = document.getElementById("category").value;

let icon = document.getElementById("icon").value;

let notes = document.getElementById("notes").value;

//obligatie

if (!productName || !expiryDate || !quantity || !category ) {

alert("Please fill out; name, quantity, expiry date and category!");

return;

}

let product = { productName, expiryDate, quantity, category, icon, notes };

//nosuta

fetch("/api/fridge-products", {

method: "POST",

headers: { "Content-Type": "application/json" },

body: JSON.stringify(product),

})

.then((response) => {

if (response.ok) {

return response.json();

} else {

throw new Error("Could not add product");

}

})

.then((data) => {

fetchProducts();

document.getElementById("productForm").reset();

showView("mainView");

})

.catch((err) => {

console.error(err);

alert("Kļūda: " + err.message);

});

});

//dzesana id

function deleteProduct(productId) {

fetch(`/api/fridge-products/${productId}`, { method: "DELETE" })

.then((response) => {

if (response.ok) {

fetchProducts();

} else {

throw new Error("Could not delete product");

}

})

.catch((err) => console.error("ERROR delete:", err));

}

//skati

function showView(view) {

document.getElementById("mainView").style.display = "none";

document.getElementById("addView").style.display = "none";

document.getElementById("editView").style.display = "none";

document.getElementById(view).style.display = "block";

}

//terminu krasas

function updateProductColor(element, expiryDate) {

let expiry = new Date(expiryDate);

let today = new Date();

let timeDiff = expiry - today;

let daysLeft = timeDiff / (1000 \* 60 \* 60 \* 24);

if (daysLeft <= 3) {

element.style.backgroundColor = "#f2b0b0";

element.style.color = "#4d4d4d";

} else if (daysLeft <= 7) {

element.style.backgroundColor = "#f2d4b0";

element.style.color = "#4d4d4d";

}

}

document.addEventListener("DOMContentLoaded", highlightExpiringProducts);

style.css

@import url('https://fonts.googleapis.com/css2?family=Poppins:ital,wght@0,100;0,200;0,300;0,400;0,500;0,600;0,700;0,800;0,900;1,100;1,200;1,300;1,400;1,500;1,600;1,700;1,800;1,900&display=swap');

:root {

--accent-color: #254b6e;

--base-color: white;

--text-color: #0d0351;

--input-color: #f2f2f2;

--warning-color: #f06272;

--border-radius: 8px;

}

\* {

margin: 0;

padding: 0;

box-sizing: border-box;

}

html {

font-family: Poppins, Segoe UI, sans-serif;

font-size: 12pt;

color: var(--text-color);

text-align: center;

}

body {

min-height: 100vh;

background-image: url(images/6016317.jpg);

background-size: cover;

background-position: right;

overflow: hidden;

}

.wrapper {

background-color: var(--base-color);

height: 100vh;

width: max(40%, 600px);

padding: 10px;

border-radius: 0 20px 20px 0;

display: flex;

flex-direction: column;

align-items: center;

justify-content: center;

}

.container {

display: flex;

flex-direction: column;

align-items: center;

padding: 20px;

}

h1 {

font-size: 3rem;

font-weight: 900;

text-transform: uppercase;

}

a {

text-decoration: none;

color: var(--accent-color);

}

a:hover {

text-decoration: underline;

}

form {

width: min(400px, 100%);

margin: 20px auto 50px;

display: flex;

flex-direction: column;

align-items: center;

gap: 15px;

}

form > div {

width: 100%;

display: flex;

justify-content: center;

}

form label {

flex-shrink: 0;

height: 50px;

width: 50px;

background-color: var(--accent-color);

fill: var(--base-color);

color: var(--base-color);

border-radius: 10px 0 0 10px;

display: flex;

justify-content: center;

align-items: center;

font-size: 1.5rem;

font-weight: 500;

}

form input,

form select,

form textarea {

flex-grow: 1;

height: 50px;

padding: 1em;

font: inherit;

border-radius: 0 10px 10px 0;

border: 2px solid var(--input-color);

border-left: none;

background-color: var(--input-color);

transition: 150ms ease;

}

form input:hover,

form select:hover,

form textarea:hover {

border-color: var(--accent-color);

}

form input:focus,

form select:focus,

form textarea:focus {

outline: none;

border-color: var(--text-color);

}

div:has(input:focus) > label {

background-color: var(--text-color);

}

form input::placeholder {

color: var(--text-color);

}

form button {

margin-top: 10px;

border: none;

border-radius: 1000px;

padding: 0.85em 4em;

background-color: var(--accent-color);

color: var(--base-color);

font: inherit;

font-weight: 600;

text-transform: uppercase;

cursor: pointer;

transition: 150ms ease;

}

form button:hover,

form button:focus {

background-color: var(--text-color);

}

.input-group {

display: flex;

align-items: center;

width: 100%;

}

.input-group label {

width: 50px;

height: 50px;

background-color: var(--accent-color);

color: var(--base-color);

border-radius: 10px 0 0 10px;

display: flex;

justify-content: center;

align-items: center;

font-size: 1.5rem;

font-weight: 500;

}

.input-group svg {

width: 24px;

height: 24px;

fill: var(--base-color);

}

.input-group input,

.input-group select,

.input-group textarea {

flex-grow: 1;

height: 50px;

padding: 1em;

font: inherit;

border-radius: 0 10px 10px 0;

border: 2px solid var(--input-color);

border-left: none;

background-color: var(--input-color);

transition: 150ms ease;

}

.input-group textarea {

height: 100px;

resize: vertical;

}

.input-group input:hover,

.input-group select:hover,

.input-group textarea:hover {

border-color: var(--accent-color);

}

.input-group input:focus,

.input-group select:focus,

.input-group textarea:focus {

outline: none;

border-color: var(--text-color);

}

.input-group:has(input:focus, select:focus, textarea:focus) > label {

background-color: var(--text-color);

}

.error-message,

#error-message {

color: var(--warning-color);

font-size: 14px;

font-weight: bold;

margin-top: 10px;

}

.grid-container {

display: grid;

grid-template-columns: repeat(auto-fit, minmax(150px, 1fr));

gap: 1rem;

padding: 1rem;

}

.grid-item {

background-color: var(--base-color);

border: 2px solid var(--input-color);

border-radius: var(--border-radius);

padding: 1rem;

min-height: 150px;

}

@media (max-width: 600px) {

.grid-container {

grid-template-columns: 1fr;

gap: 5px;

padding: 5px;

}

.grid-item {

padding: 10px;

font-size: 14px;

}

.product-box {

width: 100%;

max-width: 100%;

padding: 10px;

}

}

@media (max-width: 768px) {

.button-container {

flex-direction: column;

gap: 5px;

}

button,

.logout-button {

font-size: 14px;

padding: 8px 15px;

}

}

@media (max-width: 600px) {

form {

width: 90%;

padding: 15px;

}

input,

select,

textarea {

font-size: 16px;

padding: 12px;

}

}

@media (max-width: 600px) {

.category-container {

max-height: 200px;

}

}

@media (max-width: 600px) {

.icon {

width: 30px;

height: 30px;

}

}

.grid-item {

padding: 10px;

font-size: 14px;

}

.product-box {

width: 100%;

max-width: 100%;

padding: 10px;

}

button,

.login-button,

.category {

padding: 12px 30px;

border: none;

border-radius: 1000px;

background-color: var(--accent-color);

color: var(--base-color);

font-size: 1rem;

font-weight: 600;

cursor: pointer;

transition: background 0.2s;

}

button:hover,

.category:hover {

background-color: var(--text-color);

}

.categories {

display: flex;

flex-wrap: wrap;

gap: 10px;

margin-top: 20px;

justify-content: center;

}

.products {

width: 100%;

display: flex;

flex-direction: column;

gap: 10px;

margin-top: 20px;

align-items: flex-start;

}

.product-card {

display: flex;

align-items: center;

width: 100%;

background-color: var(--input-color);

padding: 10px 15px;

border-radius: var(--border-radius);

border: 2px solid var(--input-color);

transition: border 0.2s;

}

.product-card:hover {

border-color: var(--accent-color);

}

.edit-mode .product-card {

position: relative;

}

.product-box {

position: relative;

width: 100%;

max-width: 250px;

padding: 10px;

margin: 10px;

background-color: #f9f9f9;

border: 1px solid var(--input-color);

border-radius: var(--border-radius);

display: flex;

flex-direction: column;

align-items: center;

justify-content: space-between;

min-height: 120px;

}

.box {

display: flex;

flex-direction: column;

align-items: center;

justify-content: center;

width: 100%;

padding: 10px;

min-height: 100px;

text-align: center;

}

.product-content {

width: 100%;

word-wrap: break-word;

overflow-wrap: break-word;

}

.delete-button {

position: absolute;

top: 5px;

right: 5px;

width: 30px;

height: 30px;

background-color: #eca5ad;

border: none;

border-radius: 50%;

display: flex;

align-items: center;

justify-content: center;

cursor: pointer;

font-size: 16px;

color: white;

}

.icon {

width: 50px;

height: 50px;

background-size: cover;

background-repeat: no-repeat;

background-position: center;

}

.logout-button {

position: fixed;

bottom: 20px;

right: 20px;

padding: 12px 30px;

border: none;

border-radius: 1000px;

background-color: var(--accent-color);

color: var(--base-color);

font-size: 1rem;

font-weight: 600;

cursor: pointer;

transition: background 0.2s;

}

.category-container {

max-height: 300px;

overflow-y: auto;

padding: 10px;

display: flex;

flex-direction: column;

gap: 10px;

}

server.js

//moduli

const express = require("express");

const session = require("express-session");

const sqlite3 = require("sqlite3").verbose();

const app = express();

//css, js, atteli

app.use(express.static('public'));

//middleware

app.use(express.json());

app.use(express.urlencoded({ extended: true }));

//sesijas

app.use(

session({

secret: "fridge",

resave: false,

saveUninitialized: false,

})

);

//savienojums ar datubazi

const db = new sqlite3.Database("./users.db", (err) => {

if (err) {

console.error(err.message);

}

console.log("Connected to the users database.");

});

//izmanto ejs

app.set("view engine", "ejs");

app.use(express.static("public"));

//autentifikacija

function isAuthenticated(req, res, next) {

if (req.session && req.session.userId) {

next();

} else {

res.redirect("/");

}

}

//logij

app.get("/", (req, res) => {

res.render("login", { error: null });

});

//signup

app.get("/signup", (req, res) => {

res.render("signup", { error: null, formData: null });

});

//pieteiksanas

app.post("/logindone", (req, res) => {

const email = req.body.email;

const password = req.body.password;

db.get('SELECT \* FROM dati WHERE email = ?', [email], (err, row) => {

if (err) {

console.error(err.message);

res.status(500).send('Internal server error');

} else if (!row) {

res.render("login", { error: "E-MAIL NOT FOUND" });

} else {

bcrypt.compare(password, row.password, (err, result) => {

if (err) {

console.error(err.message);

return res.status(500).send("Internal server error");

}

if (result) {

req.session.userId = row.id;

res.redirect("/index");

} else {

res.render("login", { error: "WRONG PASSWORD" });

}

});

}

});

});

const bcrypt = require('bcrypt');

//reģistrācija

app.post("/register", (req, res) => {

const data = req.body;

console.log(data);

if (data.password !== data.repeat\_password) {

return res.render("signup", { error: "PASSWORDS DO NOT MATCH", formData: data });

}

db.get("SELECT \* FROM dati WHERE email = ?", [data.email], (err, row) => {

if (err) {

console.error(err.message);

return res.status(500).send("Internal server error");

} else if (row) {

return res.render("signup", { error: "E-MAIL IS ALREADY TAKEN", formData: data });

} else {

bcrypt.hash(data.password, 10, (err, hashedPassword) => {

if (err) {

console.error(err.message);

return res.status(500).send("Internal server error");

}

const sql = 'INSERT INTO dati(firstname, email, password) VALUES (?, ?, ?)';

db.run(sql, [data.firstname, data.email, hashedPassword], function (err) {

if (err) {

console.error(err.message);

return res.status(500).send("Internal server error");

} else {

res.redirect("/");

}

});

});

}

});

});

//iziešana

app.get("/logout", (req, res) => {

req.session.destroy();

res.redirect("/");

});

//ledusskapja skats

app.get("/index", isAuthenticated, (req, res) => {

res.render("index", {error:null});

});

// api lietotaja produkti

app.get("/api/fridge-products", isAuthenticated, (req, res) => {

const userId = req.session.userId;

const sql = "SELECT \* FROM fridge\_products WHERE user\_id = ?";

db.all(sql, [userId], (err, rows) => {

if (err) {

console.error(err.message);

res.status(500).json({ error: "Internal server error" });

} else {

res.json(rows);

}

});

});

//iegust produktu

app.get('/api/products', async (req, res) => {

try {

const products = await db.all("SELECT id, name, icon FROM products");

res.json(products);

} catch (error) {

res.status(500).json({ error: "Database error" });

}

});

//ievieno produktus

app.post("/api/fridge-products", isAuthenticated, (req, res) => {

const userId = req.session.userId;

const { productName, expiryDate, quantity, category, icon, notes } = req.body;

if (!productName || !expiryDate || !quantity || !category || !icon) {

return res.status(400).json({ error: "PLEASE FILL OUT PRODUCT NAME, EXPIRY DATE, QUANTITY, CATGEORY AND ICON" });

}

const sql =

"INSERT INTO fridge\_products (user\_id, product\_name, expiry\_date, quantity, category, icon, notes) VALUES (?, ?, ?, ?, ?, ?, ?)";

db.run(sql, [userId, productName, expiryDate, quantity, category, icon, notes], function (err) {

if (err) {

console.error(err.message);

res.status(500).json({ error: "Internal server error" });

} else {

res.status(201).json({

id: this.lastID,

user\_id: userId,

product\_name: productName,

expiry\_date: expiryDate,

quantity,

category,

icon,

notes,

});

}

});

});

//dzeš

app.delete("/api/fridge-products/:id", isAuthenticated, (req, res) => {

const userId = req.session.userId;

const productId = req.params.id;

const sqlCheck = "SELECT \* FROM fridge\_products WHERE id = ? AND user\_id = ?";

db.get(sqlCheck, [productId, userId], (err, row) => {

if (err) {

console.error(err.message);

return res.status(500).json({ error: "Internal server error" });

} else if (!row) {

return res.status(404).json({ error: "PRODUCT NOT FOUND" });

} else {

const sqlDelete = "DELETE FROM fridge\_products WHERE id = ?";

db.run(sqlDelete, [productId], function (err) {

if (err) {

console.error(err.message);

res.status(500).json({ error: "Internal server error" });

} else {

res.status(204).send();

}

});

}

});

});

//serveris

const port = process.env.PORT || 3000;

app.listen(port, '0.0.0.0', () => {

console.log(`Server is running on port ${port}`);

});