

Dokumentācija

Programmēšanā

Veidoja: Edvards Klinklāvs

Rīgas Tālmācības vidusskola

2024./2025. m. g.

Satura rādītājs

Dokumentācija	1
1. Problēmas izpēte un analīze	3
2. Programmatūras prasību specifikācija	4
3. Programmatūras izstrādes plāns.....	8
4. Atklūdošanas un akcepttestēšanas pārskats	9
5. Lietotāja ceļvedis	10
6. Piemērotās licences pamatojums	12
7. Pielikums (Programmatūras kods).....	13

1. Problēmas izpēte un analīze

Ikdienā jau sen novēroju, ka manā ģimenē bieži tiek kaut kas aizmirsts, nokavēts, sajaukti apmeklējumu datumi, aizmirsts apsveikt kādu radnieku. Lai arī maniem vecākiem ir viedtālruni ar funkcijām, kā, piemēram, Kalendārs, Atgātnes, viņi uzskata, ka tie nav ērti lietošanā un tos neizmanto, bet tā vietā raksta uz lapiņām sarakstus un dienai nepieciešamo informāciju ar roku. Šāds plānojuma pieraksts nav efektīvs un ērts. Savukārt jau gatavās programmas, kas pieejamas internetā, nav pielāgotas un tik vienkārši lietojamas. Novēroju arī, ka mana mamma vienu gadu bija nopirkusi plānotāju grāmatnīcā. Viņa to no sākuma pildīja un visur nēsāja līdzi, bet pēc kāda laika pārgāja atkal uz lapiņu sistēmu, rakstot sarakstus garākam periodam ar atgādinājumiem un tad pārrakstot katru dienu atkal jaunus sarakstus jau konkrētajai dienai.

Uzzinot par šo projektu, man uzreiz ienāca prātā ideja padiskutēt ar ģimeni par novēroto problēmu un potenciālajiem risinājumiem, kā to varētu uzlabot vai novērst. Es veicu aptauju ar savu ģimeni, ar katru ģimenes locekli atsevišķi, lai uzzinātu precīzāk katra ģimenes locekļa problēmas šajā jautājumā, vajadzības un variantus, kāds risinājums viņam būtu noderīgs. Es aptaujāju mammu, tēti un vecmammu. Aptaujājot savus ģimenes locekļus, es veicu pierakstus un secināju, ka katram ģimenes loceklim ir atšķirīgas lietas, ko viņi pieraksta vai ko viņiem nepieciešams atcerēties, kā, piemēram,

- **Mammai**- svarīgi pierakstīt visus datumus, kad manam mazajam brālim un mātai ir kādas skolas aktivitātes, vecāku sapulces, dakteru vizītes visai ģimenei, tāpat arī svarīgi darba pasākumi un arī jubilejas;
- **Tētim**- svarīgi pierakstīt plānotās darba tikšanās ar klientiem, lai nesaplānotu vienā datumā vairākas, kas savā starpā pārklātos, tāpat arī viņš parasta pieraksta klāt vēl kādu nozīmīgu informāciju, kas svarīga priekš tikšanās. Pēc tikšanās viņš parasti saraksta, kad viņam jāuzsāk darbs pie klienta, kas jādara, kad jāveic materiālu iegāde, u.c. biznesa pieraksti par katru klientu un viņa projektu;
- **Vecmammai**- svarīgi pierakstīt visas jubilejas, ārstu pierakstus un viņa gribētu, ka varētu saplānot arī nedēļas ēdienkarti veselīgai diētai;
- **Man pašam**- svarīgi pierakstīt mēneša mācību plānu ar termiņiem, un arī ikdienas plānu gan skolas vajadzībām, gan darbam un treniņiem.

Ar visiem ģimenes locekļiem pārrunāju iespējamus risinājumus manu iespēju robežās, kas viņiem būtu ērti, lai datorizētu viņu pierakstus. Visi piekrita, ka tas būtu noderīgi pie nosacījuma, ka visu pierakstīto informāciju var izprintēt.

Mani secinājumi bija, ka, lai visi lietotāji būtu apmierināti, sistēmai jābūt vienkārši pieejamai, viegli pārskatāmai, viegli aizpildāmai, labojamai un printējamai.

2. Programmatūras prasību specifikācija

Mērķauditorija būs visi manas mājas iedzīvotāji (Visi kas spēj pārvietoties internetā), jo serveris strādās uz lokālā interneta tīkla un būs pieejams mājvietas iedzīvotājiem.

Sistēma strādās uz Flask servera, lai mājas iedzīvotāji ar lokālo IP adresi var pieslēgties serverim, izmantojot vietējo internetu, kas padarīs programmu pieejamu mājas iedzīvotājiem jebkurā brīdī.

Sistēmā vajadzēs realizēt:

1. Sākuma lapu – lapa ar nosaukumu, kas izskaidro programmas mērķi un poga, kas ved lietotāju uz ielogošanos un reģistrēšanās lapu.

Sākuma lapas skice:



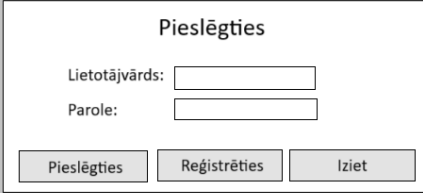
2. Reģistrēšanās lapu, kas ļaus katram jaunam lietotājam izveidot savu personīgo accountu (pašreģistrācija), ar kuru pēc tam piekļūtu pie sistēmas. Šajā lapā arī būs divi ievades logi, viens priekš lietotājevārda un otrs priekš paroles, kā arī šajā lapā atradīsies 3 pogas, viena, kas aizvedīs lietotāju uz sākuma lapu, otra, kas ļaus pārvietoties no reģistrācijas lapas uz ielogošanās lapu, un trešā, kas saglabās lietotāja datus datubāzē un atgriezīs lietotāju uz ielogošanās lapu pēc jauna konta izveides.

Reģistrēšanās lapas skice:

The sketch shows a registration form titled "Reģistrēties". It contains two input fields: "Lietotājevārds:" and "Parole:". Below these fields are two buttons: "Reģistrēties" and "Atpakaļ uz pieslēgšanos".

3. Pieslēgšanās lapu, kas ir svarīga, lai lietotāji varētu tikt sistēmā, jo bez tās nevar. Šajā lapā arī būs divi ievades logi, viens priekš lietotājevārda un otrs priekš paroles, kā arī šajā lapā atradīsies 3 pogas, pirmā, kas atgriezīs lietotāju uz sākuma lapu, otrā, kas aizved lietotāju no ielogošanās uz reģistrēšanās lapu, un trešā, kas pārbauda lietotāja konta datus un ielaiž to sistēmā.

Pieslēgšanās lapas skice:



The sketch shows a login form with the title "Pieslēgties". It includes two text input fields: "Lietotājsvārds:" and "Parole:". Below these fields are three buttons: "Pieslēgties", "Reģistrēties", and "Iziet".

4. Galvenā lapas viena puse, sānamala, kurā atrodas 4 pogas :
 - a. Pirmais (Rediģēt) ieslēdz un izslēdz rediģēt funkciju, kas ļauj dzēst un pievienot jaunus atgādinājumus.
 - b. Otrā poga (Svarīgi) ieslēdz un izslēdz atlasīšanu, kas atlasa tikai svarīgos ar zvaigznīti atzīmētos atgādinājumus, trešais un 1 poga, kā arī autora nospiedums.
 - c. Trešā poga (Par aplikāciju) parāda un noslēpj autora nospiedumu un projekta izveides sākuma un beigu datumus.
 - d. Ceturtā poga (Atslēgties) atvieno lietotāju un aizved uz sākuma lapu.
5. Galvenās lapas otra puse, kur atrodas tabula ar tabulas virsrakstu un ar tās atgādinājumiem. Tabulā būs 4 ailes:
 - a. "Uzdevums", kur tiks ierakstīts atgādinājums.
 - b. "Termiņš", kurā tiks ievadīts atgādinājuma noteiktais termiņš.
 - c. "Progress", kur varēs nomainīt atgādinājuma progresu : neiesākts, iesākts, nokavēts un pabeigts.
 - d. "Svarīgi", kurā varēs atzīmēt ar zvaigznīti, vai ir svarīgs atgādinājums un pēc tam tos var atlasīt ar pogu (Svarīgi).

Galvenās lapas skices:

Rediģēt

Svarīgi

Par aplikāciju

Atslēgties

Par aplikāciju

autors: _____

Projekta sākums: ____

Projekta beigšanas datums _____

Atgādinājumi

uzdevums	Termiņš	Progress	Svarīgi
Konkrēta lieta kas jāizdara	29.02.2025	progresā	★

Rediģēt

Svarīgi

Par aplikāciju

Atslēgties

Par aplikāciju

autors: _____

Projekta sākums: ____

Projekta beigšanas datums _____

Atgādinājumi

Jauns atgādinājums

mm/dd/yyyy

Pievienot

uzdevums	Termiņš	Progress	Svarīgi	Darbība
Konkrēta lieta kas jāizdara	29.02.2025	progresā	★	dzēst

Sistēmā tiks ieviestas tikai pašas svarīgākās funkcijas. Tās sistēmu padarītu pārskatāmu, viegli lietojamu, kas samazinātu neskaidrības un padarītu sistēmu kompaktāku un vienkāršāku lietotājam. Papildus mājas lapai būs pieejama printēšana, kas ir iebūvēta gandrīz visās meklētājprogrammās.

3. Programmatūras izstrādes plāns

Projekta veidošanai izmantošu ātrās prototipēšanas metodi, jo, manuprāt, visi mani projekti tiek veidoti, balstoties uz šo pieeju. Kā piemēram, projektēšanas procesā vispirms izveidoju kādu funkciju un to pārbaudu, pēc tam piestrādāju pie izskata, un vēlāk pievienoju vēl kādas jaunas funkcionalitātes. Šāda pieeja ļauj man efektīvi attīstīt projektu un pielāgot to nepieciešamajām prasībām.

Projekta izstrādes plāns:

1. Izveidot lietotāja saskarsnes dizainu (sākuma, reģistrēšanās, pieslēgšanās un galveno lapu) un projekta struktūru.
2. Izstrādāt datubāzes struktūru.
3. Izstrādāt reģistrēšanās un pieslēgšanās funkcijas un uzstādīt bcrypt šifrēšanu.
4. Reģistrēšanās un pieslēgšanās funkciju testēšana un labošana.
5. Izveidot galvenajā lapā atgādinājumu pievienošanas un dzēšanas funkcijas.
6. Izveidot funkcijas pogām (svarīgi, par aplikāciju un atslēgties).
7. Izstrādāt pogām ieslēgšanas un izslēgšanas (slēdža) sistēmu.
8. Galvenā sistēmas funkcionalitātes testēšana.
9. Dokumentācijas un lietotāja rokasgrāmatas izveide.
10. Publicēšana.

Projekta nākotnes uzlabojumi:

1. Ieviest lietotājvārda un paroles atgūšanas iespējas.
2. Ieviest vairāk funkcijas saistībā ar atgādinājumu šķirošanu, atlasīšanu.
3. Uzlabot vispārējo dizainu, kā piemēram: padarīt to modernāku, pievilcīgāku.
4. Palaist projektu kā pilnu mājas lapu, lai pie tās var piekļūt no jebkuras vietas ar internetu.
5. Ieviest funkciju, ka var rediģēt jau ievietotos atgādinājumus.

4. Atklūdošanas un akcepttestēšanas pārskats

Programmu atklāju pats, pārbaudot katru iecerēto funkciju, iedevu patestēt potenciāliem lietotājiem un ievācu atsaucis pēc kurām salaboju, uzlaboju programmatūru.

Funkcijas kas tika pārbaudītas:

1. Ielogošanās un izlogošanās (minimālais simbolu skaits, vienādi lietotājevārdi un paroles šifrēšana)

Minimālais simbolu skaits- Šo funkciju es pievienoju un pārbaudīju lai lietotāji varētu izveidot drošākas paroles un lai citiem nebūtu tik viegli piekļūt taviem personīgajiem datiem.

```
<input type="password" id="password" name="password" minlength="6" required>
```

Vienādi lietotājevārdi - No sākuma, kad izmantoju vienādu lietotājevārdu, pa visu mājas lapu izmet error, bet ar šo labojumu tas mājaslapā parādīja tekstu, ka lietotājs eksistē un novērsa kļūdu.

except sqlite3.IntegrityError:

```
return render_template('register.html', texts=TEXTS,
error=TEXTS['error_user_exists'])
```

Paroles šifrēšana - Ieviešot un testējot paroles šifrēšanu sastapos ar kļūdu:

AttributeError: 'bytes' object has no attribute 'encode'. Did you mean: 'decode'? Šī kļūda radās, kad ieviesu sistēmā bcrypt. Funkcijā login konstatēju, ka parole jau bija iekriptēta bitos un tāpēc sistēma meta kļūdu.

```
@app.route('/login', methods=['GET', 'POST'])
def login():
    if request.method == 'POST':
        username = request.form['username']
        password = request.form['password'].encode('utf-8')

        with sqlite3.connect('data.db') as conn:
            cursor = conn.cursor()
            cursor.execute('SELECT * FROM users WHERE username = ?', (username,))
            user = cursor.fetchone()

            if user and bcrypt.checkpw(password.encode('utf-8'), user[2]):
                session['user_id'] = user[0]
                session['username'] = user[1]
                return redirect(url_for('dashboard'))
            else:
                return render_template('login.html', texts=TEXTS, error=TEXTS['error_invalid_credentials'])

    return render_template('login.html', texts=TEXTS)
```

2. Jauna atgādinājuma izveide (teksta iepilde, pārbaudu logu max. izplešanos, pogu slēgāšana, lapas samazināšana un palielināšana)

Teksta iepilde, logu max. izplešanās: Risinātas bija daudzas problēmas, veikti uzlabojumi no lietotāju ieteikumiem, kā, piemēram:

pēc lietotāja atsaucies ievietojot jaunu atgādinājuma tekstu `<input>` sadaļā, sadaļa palika tādā pašā izmērā un padarīja lielu sarakstu rakstīšanu ļoti sarežģītu. Pats arī biju pamanījis šo problēmu, ko salaboju, nomainot `<input>` pret `<textarea>` kas ļauj palielināt ievades logu. Manuprāt šis izskatījās labāk un saprotamāk, uzlabojot vispārējo programmas kopskatu.

Lapas samazināšana un palielināšana: Šis tika pārbaudīts, lai lietotāji ar sliktu redzi vai cilvēki, kuriem patīk lielāks fonts, varētu mājas lapu palielināt un lai visi objekti lapā neaizlidotu, kur vien patīk. Šo var novērst, izmantojot css failā izmantoju procentus priekš objektu novietošanas nevis pixelus, jo tie, palielinot lapu, neietekmēs objekta novietojumu.

5. Lietotāja ceļvedis

Programmas uzstādīšana :

Lai palaistu šo servera programmu lietotājam vajag aiziet uz python mājaslapu un nolādēt vēlams jaunāko python versiju, tad uzinstalējot aplikāciju atzīmēt ķeksīti pie (Add to Path) kas pie instalēšanas pievienos "Python" un "Pip" pie (Windows Powershell) aplikācijas.

Tālāk nepieciešams ielādēt flask un bcrypt, lai to izdarītu, vajag atvērt "Windows Powershell" un ievadīt sekojošas 2 komandas:

1. pip install flask
2. pip install bcrypt

Lai uzsāktu programmu failus vajag (extract) no zip mapes un novietot labi zināmā vietā, lai sistēma varētu atrast lokāciju.

Jānokopē atrašanās vieta līdzīgi kā šeit:

```
C:\Users\Edvards\Desktop\vid.prog.darb\dzīvesvietas.lietotāju.plānošanas.ierīce
```

```
C:\Users\Edvards\Desktop\vid.prog.darb\dzīvesvietas.lietotāju.plānošanas.ierīce
```

Tālāk jāieiet (Windows Powershell), jāuzraksta cd un aiz tā jāielīmē atrašanās vieta, tā, lai tā izskatītos līdzīgi šai:

```
PS C:\Users\Edvards> cd Desktop\vid.prog.darb\dzīvesvietas.lietotāju.plānošanas.ierīce
```

Tad jāpalaiž šo funkciju: python appy.py

```
PS C:\Users\Edvards\Desktop\vid.prog.darb\dzīvesvietas.lietotāju.plānošanas.ierīce> python appy.py
```

Tagad viss ir ieslēgts un ar to (http://) linku var apmeklēt šo serveri.

Lai pieslēgtos no citas ierīces, serveris jāpalaiž ar komandu

```
[ python appy.py --host=0.0.0.0 --port=5000 ]
```

Iespējams, ka "Windows Defender Firewall" vajag atļaut python piekļuvi, ko var izdarīt sānā "Allow apps to communicate thorough Windows Defender Firewall".

Un, lai pieslēgtos, jāieraksta (Windows Powershell) aplikācijā [ipconfig]

Zem (Wireless LAN adapter Wi-Fi:) jāatrod IPv4 Address un jāievieto: http://_____:5000 un tas būs links uz mājaslapu.

6. Piemērotās licences pamatojums

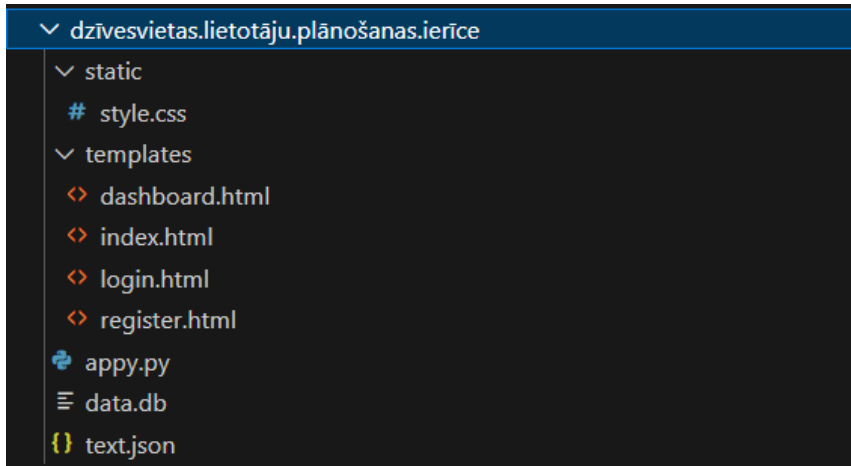
Flask : Licencēs nav nekādu aizliegumu un viss ir par brīvu.

Mana produkta licence:

1. Visi potenciālie lietotāji manu programmu var lietot par brīvu, nevienam nav jāmaksā.
2. Programmas izejas kodu un visu pārējo var rediģēt vai mainīt pēc paša vajadzībām.
3. Šo programmas kodu var kopēt un izmantot visām savām vajadzībām.

7. Pielikums (Programmatūras kods)

Programmas failu struktūra:



Programmas pilnais kods:

```
<> register.html X
dzīvesvietas.lietotāju.plānošanas.ierīce > templates > <> register.html
1  <!DOCTYPE html>
2  <html lang="lv">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width, initial-scale=1.0">
6      <link rel="stylesheet" href="static/style.css">
7      <title>Reģistrācija</title>
8  </head>
9  <body class="startbox">
10     <h1>Reģistrēties</h1>
11     {% if error %}
12         <p style="color: red;">{{ error }}</p>
13     {% endif %}
14     <form method="POST" action="{{ url_for('register') }}">
15         <label for="username">Lietotājvārds:</label>
16         <input type="text" id="username" name="username" minlength="4" required>
17         <br>
18         <label for="password">Parole:</label>
19         <input type="password" id="password" name="password" minlength="6" required>
20         <br>
21         <div class="buttons">
22             <button class="but" type="submit">Reģistrēties</button>
23         </div>
24         <a class="but" href="{{ url_for('login') }}">Atpakaļ uz pieslēgšanos</a>
25         <br>
26         <br>
27     </div>
28 </body>
29 </html>
30
```

<> index.html X

dzīvesvietas.lietotāju.plānošanas.ierīce > templates > <> index.html

```
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <link rel="stylesheet" href="static/style.css">
5      <title>{{ texts.title }}</title>
6  </head>
7  <body class="startbox">
8      <h1>{{ texts.name }}</h1>
9      <a class="but" href="{{ url_for('login') }}">Sākums</a>
10 </body>
11 </html>
12
```

<> login.html X

<> register.html

dzīvesvietas.lietotāju.plānošanas.ierīce > templates > <> login.html

```
1  <!DOCTYPE html>
2  <html lang="lv">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width, initial-scale=1.0">
6      <link rel="stylesheet" href="static/style.css">
7      <title>Pieslēgties</title>
8  </head>
9  <body class="startbox">
10     <h1>Pieslēgties</h1>
11     {% if error %}
12     <p style="color: red;">{{ error }}</p>
13     {% endif %}
14
15     <form method="POST" action="{{ url_for('login') }}">
16         <label for="username">Lietotājvārds:</label>
17         <input type="text" id="username" name="username" minlength="4" required>
18         <br>
19         <label for="password">Parole:</label>
20         <input type="password" id="password" name="password" minlength="6" required>
21         <br>
22     <div class="buttons">
23         <button class="but" type="submit">Pieslēgties</button>
24     </form>
25     <a class="but" href="{{ url_for('register') }}">Reģistrēties</a>
26     <a class="but" href="{{ url_for('index') }}">Iziet</a>
27
28 </div>
29 <br>
30 </body>
31 </html>
32
```

```
# style.css X
dzīvesvietas.lietotāju.plānošanas.ierīce > static > # style.css > ...
1  #sidebar{
2      border: 2px solid black;
3      border-radius: 5px;
4      background-color: white;
5      width: 20%;
6      float: left;
7      height: 200px;
8  }
9
10 #content{
11     border: 2px solid black;
12     border-radius: 5px;
13     background-color: lightgray;
14     width: 78%;
15     float: right;
16     height: 100%;
17     min-height: 600px;
18     padding-bottom: 100px;
19 }
20
21 #maintable{
22     width: 65%;
23     height: 100%;
24     text-align: center;
25     font-size: 100%;
26     margin-left: 15%;
27     margin-right: 15%;
28 }
29
30 table {
31     border-collapse: collapse;
32     width: 100%;
33     margin-top: 2%;
34 }
35
36 th, td {
37     border: 2px solid black;
38     padding: 8px;
39     text-align: left;
40     word-wrap: break-word;
41     max-width: 100px;
42     background-color: rgb(192, 192, 192);
43 }
44
45 th {
46     background-color: rgb(151, 151, 151);
47 }
48
49 html{
50     background-color: rgb(189, 189, 189);
51 }
```

```
51 }
52
53 .buttons{
54     padding-top: 5px;
55 }
56
57 .but{
58     animation: none;
59     color: black;
60     text-decoration: none;
61     margin: 10px;
62     font-size: 16px;
63     text-align: center;
64     border: 2px solid black;
65     border-radius: 3px;
66     background-color: white;
67     margin: 2px;
68     padding: 2px;
69     font-size: 18px;
70     font-family: Arial;
71 }
72
73 .startbox{
74     width: 25%;
75     height: 100%;
76     border: 2px solid black;
77     border-radius: 5px;
78     background-color: azure;
79     text-align: center;
80     margin-left: 37.5%;
81     margin-right: 37.5%;
82     padding-bottom: 10px;
83 }
84
85 #password{
86     margin-left: 11.5%;
87     margin-top: 1%;
88 }
89
90 .bar{
91     width: 100%;
92     height: 25%;
93     font-size: 20px;
94     animation: none;
95     text-decoration: none;
96 }
97
98 }
```

```


98
99 #redpoga{
100     border: 2px solid □black;
101     padding: 8px;
102     background-color: ■white;
103     position: relative;
104     top: 80%;
105     left: 62%;
106     border-radius: 5px;
107     animation: none;
108     text-decoration: none;
109     color: □black;
110     font-family: Arial;
111     font-size: 16px;
112 }
113
114 #About{
115     border: 2px solid □black;
116     background-color: ■white;
117     width: 70%;
118     height: auto;
119     float: left;
120     margin-left: 5%;
121     margin-top: 5%;
122     padding: 8px;
123 }
124
125 #darb{
126     text-align: center;
127     animation: none;
128     text-decoration: none;
129     color: □black;
130     size: 30px;
131 }
132
133 #zvaiz{
134     text-align: center;
135     animation: none;
136     text-decoration: none;
137     color: ■yellow;
138     size: 30px;
139     padding-left: 45%;
140     padding-right: 45%;
141 }
142 textarea{
143     field-sizing: content;
144     max-width: 400px;
145     max-block-size: 300px;
146     width: 140px;
147     border: 1px solid □black;
148     text-align: center;
149     float: left ;
150     width: 400px;
151 }

```



```
dashboard.html X
dzīvesvietas.lietotāju.plānošanas.ierīce > templates > dashboard.html
1 <!DOCTYPE html>
2 <html>
3 <head>
4 <link rel="stylesheet" href="static/style.css">
5 <title>{{ texts.title }}</title>
6 </head>
7 <body>
8 <div id="sidebar">
9 <a href="{{ url_for('toggle_edit_mode') }}"><button class="bar">{% if edit_mode %}Beigt rediģēšanu{% else %}Rediģēt{% endif %}</button></a>
10 <a href="{{ url_for('toggle_show_important') }}"><button class="bar">{% if show_important %}Visi atgādinājumi{% else %}Svarīgi{% endif %}</button></a>
11 <a href="{{ url_for('toggle_par_info') }}"><button class="bar">Par aplikāciju</button></a>
12 <a href="{{ url_for('index') }}"><button class="bar">Atslēgties</button></a>
13
14 {% if par_info %}
15 <div id="About">
16 <h4>Par Aplikāciju </h4>
17 <a> Autors: Edvards Klinklāvs</a><br>
18 <a> Projekts sākts: 19.02.2025 </a><br>
19 <a> Un izdots: 27.02.2025</a>
20 </div>
21 {% endif %}
22 </div>
23
24 <div id="content">
25 <div id="maintable">
26 <h1>Atgādinājumi</h1>
27
28 {% if edit_mode %}
29 <form method="POST" action="{{ url_for('add_reminder') }}">
30 <textarea name="task" placeholder="Jauns atgādinājums" required></textarea>
31 <input type="date" name="due_date" required>
32 <button type="submit">Pievienot</button>
33 </form>
34 {% endif %}
35
36 <table>
37 <tr>
38 <th>Uzdevums</th>
39 <th>Terminš</th>
40 <th>Progress</th>
41 <th>Svarīgi</th>
42 {% if edit_mode %}<th>Darbība</th>{% endif %}
43 </tr>
44 {% if reminders %}
45 {% for reminder in reminders %}
46 <tr>
47 <td>{{ reminder[1] }}</td>
48 <td>{{ reminder[2] }}</td>
49 <td>
```

```
dzīvesvietas.lietotāju.plānošanas.ierīce > templates > dashboard.html
7 <body>
24 <div id="content">
25 <div id="maintable">
36 <table>
46 <tr>
49 <td>
50 <form method="GET" action="{{ url_for('update_progress', reminder_id=reminder[0]) }}">
51 <select name="progress" onchange="this.form.submit()">
52 <option style="color: green;" value="neiesākts" {% if reminder[4] == 'neiesākts' %}selected{% endif %}>neiesākts</option>
53 <option style="color: orange;" value="progresa" {% if reminder[4] == 'progresa' %}selected{% endif %}>iesākts</option>
54 <option style="color: red;" value="nokavēts" {% if reminder[4] == 'nokavēts' %}selected{% endif %}>nokavēts</option>
55 <option value="Pabeigts" {% if reminder[4] == 'Pabeigts' %}selected{% endif %}>Pabeigts</option>
56 </select>
57 </form>
58 </td>
59 <td class="funct">
60 <a id="izvai2" href="{{ url_for('toggle_important', reminder_id=reminder[0]) }}">
61 <div>
62 <div>
63 <div>
64 <div>
65 <div>
66 </div>
67 </div>
68 </div>
69 <div>
70 <div>
71 <div>
72 <div>
73 </div>
74 </div>
75 </div>
76 <div>
77 <div>
78 <div>
79 </div>
80 </div>
81 </div>
82 </div>
83 </body>
84 </html>
85
```

dzīvesvietas.lietotāju.plānošanas.ierīce >  appy.py

```

1  from flask import Flask, render_template, request, redirect, url_for, session
2  import sqlite3
3  import json
4  import bcrypt
5  from datetime import datetime
6
7  with open('text.json', 'r', encoding='utf-8') as f:
8      TEXTS = json.load(f)
9
10 app = Flask(__name__)
11 app.secret_key = 'secret_key'
12
13 def init_db():
14     with sqlite3.connect('data.db') as conn:
15         cursor = conn.cursor()
16         cursor.execute('''CREATE TABLE IF NOT EXISTS users (
17             id INTEGER PRIMARY KEY AUTOINCREMENT,
18             username TEXT UNIQUE NOT NULL,
19             password TEXT NOT NULL)
20         ''')
21
22         cursor.execute('''CREATE TABLE IF NOT EXISTS reminders (
23             id INTEGER PRIMARY KEY AUTOINCREMENT,
24             user_id INTEGER NOT NULL,
25             task TEXT NOT NULL,
26             due_date TEXT,
27             due_time TEXT,
28             progress TEXT DEFAULT 'neiesākts',
29             important INTEGER DEFAULT 0,
30             FOREIGN KEY(user_id) REFERENCES users(id))
31         ''')
32
33     conn.commit()
34
35 def update_missed_reminders():
36     with sqlite3.connect('data.db') as conn:
37         cursor = conn.cursor()
38         current_date = datetime.now().strftime('%Y-%m-%d')
39         current_time = datetime.now().strftime('%H:%M:%S')
40         cursor.execute('''
41             UPDATE reminders
42             SET progress = 'nokavēts'
43             WHERE due_date < ? OR (due_date = ? AND due_time < ?)
44             ''', (current_date, current_date, current_time))
45         conn.commit()
46
47 @app.route('/')
48 def index():
49     return render_template('index.html', texts=TEXTS)
50

```

```

50
51 @app.route('/register', methods=['GET', 'POST'])
52 def register():
53     if request.method == 'POST':
54         username = request.form['username']
55         password = request.form['password'].encode('utf-8')
56
57         hashed_password = bcrypt.hashpw(password, bcrypt.gensalt())
58
59         with sqlite3.connect('data.db') as conn:
60             cursor = conn.cursor()
61             try:
62                 cursor.execute('INSERT INTO users (username, password) VALUES (?, ?)', (username, hashed_password))
63                 conn.commit()
64                 return redirect(url_for('login'))
65             except sqlite3.IntegrityError:
66                 return render_template('register.html', texts=TEXTS, error=TEXTS['error_user_exists'])
67
68     return render_template('register.html', texts=TEXTS)
69
70 @app.route('/login', methods=['GET', 'POST'])
71 def login():
72     if request.method == 'POST':
73         username = request.form['username']
74         password = request.form['password']
75
76         with sqlite3.connect('data.db') as conn:
77             cursor = conn.cursor()
78             cursor.execute('SELECT * FROM users WHERE username = ?', (username,))
79             user = cursor.fetchone()
80
81             if user and bcrypt.checkpw(password.encode('utf-8'), user[2]):
82                 session['user_id'] = user[0]
83                 session['username'] = user[1]
84                 return redirect(url_for('dashboard'))
85             else:
86                 return render_template('login.html', texts=TEXTS, error=TEXTS['error_invalid_credentials'])
87
88     return render_template('login.html', texts=TEXTS)
89

```

```

89
90 @app.route('/dashboard')
91 def dashboard():
92     if 'user_id' not in session:
93         return redirect(url_for('login'))
94
95     update_missed_reminders()
96
97     edit_mode = session.get('edit_mode', False)
98     par_info = session.get('par_info', False)
99     show_important = session.get('show_important', False)
100
101     with sqlite3.connect('data.db') as conn:
102         cursor = conn.cursor()
103         if show_important:
104             cursor.execute('SELECT id, task, due_date, due_time, progress, important FROM reminders WHERE user_id = ? AND important = 1', (session['user_id'],))
105         else:
106             cursor.execute('SELECT id, task, due_date, due_time, progress, important FROM reminders WHERE user_id = ?', (session['user_id'],))
107         reminders = cursor.fetchall()
108
109     return render_template('dashboard.html', texts=TEXTS, reminders=reminders, edit_mode=edit_mode, par_info=par_info, show_important=show_important)
110
111 @app.route('/update_progress/<int:reminder_id>')
112 def update_progress(reminder_id):
113     if 'user_id' not in session:
114         return redirect(url_for('login'))
115
116     progress = request.args.get('progress')
117     with sqlite3.connect('data.db') as conn:
118         cursor = conn.cursor()
119
120         if progress == "Pabeigts":
121             cursor.execute('DELETE FROM reminders WHERE id = ?', (reminder_id,))
122         else:
123             cursor.execute('UPDATE reminders SET progress = ? WHERE id = ?', (progress, reminder_id))
124
125     conn.commit()
126
127     return redirect(url_for('dashboard'))
128

```

```

128
129 @app.route('/add_reminder', methods=['POST'])
130 def add_reminder():
131     if 'user_id' not in session:
132         return redirect(url_for('login'))
133
134     task = request.form['task']
135     due_date = request.form['due_date']
136     due_time = request.form.get('due_time', None)
137
138     with sqlite3.connect('data.db') as conn:
139         cursor = conn.cursor()
140         cursor.execute('INSERT INTO reminders (user_id, task, due_date, due_time) VALUES (?, ?, ?, ?)',
141                        (session['user_id'], task, due_date, due_time))
142         conn.commit()
143
144     return redirect(url_for('dashboard'))
145
146 @app.route('/delete_reminder/<int:reminder_id>')
147 def delete_reminder(reminder_id):
148     if 'user_id' not in session:
149         return redirect(url_for('login'))
150
151     with sqlite3.connect('data.db') as conn:
152         cursor = conn.cursor()
153         cursor.execute('DELETE FROM reminders WHERE id = ? AND user_id = ?', (reminder_id, session['user_id']))
154         conn.commit()
155
156     return redirect(url_for('dashboard'))
157
158 @app.route('/toggle_edit_mode')
159 def toggle_edit_mode():
160     session['edit_mode'] = not session.get('edit_mode', False)
161     return redirect(url_for('dashboard'))
162
163 @app.route('/toggle_par_info')
164 def toggle_par_info():
165     session['par_info'] = not session.get('par_info', False)
166     return redirect(url_for('dashboard'))
167

```

```

167
168 @app.route('/toggle_important/<int:reminder_id>')
169 def toggle_important(reminder_id):
170     if 'user_id' not in session:
171         return redirect(url_for('login'))
172
173     with sqlite3.connect('data.db') as conn:
174         cursor = conn.cursor()
175         cursor.execute('SELECT important FROM reminders WHERE id = ?', (reminder_id,))
176         important = cursor.fetchone()[0]
177         new_important = 0 if important else 1
178         cursor.execute('UPDATE reminders SET important = ? WHERE id = ?', (new_important, reminder_id))
179         conn.commit()
180
181     return redirect(url_for('dashboard'))
182
183 @app.route('/toggle_show_important')
184 def toggle_show_important():
185     session['show_important'] = not session.get('show_important', False)
186     return redirect(url_for('dashboard'))
187
188 if __name__ == '__main__':
189     init_db()
190     app.run(debug=True)
191

```