Home Automation

Testare

Membrii echipei:

Istrate Sebastian-Nicolae – programator șef

Ionescu Sergiu-Marian – asistent șef

Moticica Vlad-Florin – secretar

Ionică Florentina-Bianca

Matei Georgiana

Ludică Maria-Alexandra

Mihai Andrei

Negriu Andrei-Valentin

Cuprins

[1. Memoriu tehnic 3](#_7ol8fif046yf)

[2. Testarea codului 4](#_k87040kmu2y5)

[3. Mijloace de verificare 24](#_tn5aegoepoeh)

[4. Condiții de funcționare 26](#_h4lm3u7trqcp)

# 1. Memoriu tehnic

Odată cu trecerea în 2023, au apărut o multitudine de noutăți în domeniul tehnologic, iar una dintre cele mai folosite tehnologii din rândul proprietarilor de locuințe este home automation.

În acest sens, prezentul proiect a demarat în data de 16 martie 2023 prin prezentarea unei Teme tehnice la sediul companiei WiseBee Technologies din Timișoara de pe Bulevardul Vasile Pârvan 2, de către domnul inginer Ionescu Gheorghe, reprezentantul SoftDeliver Technologies.

Solicitarea în cauză vizează implementarea unui sistem de automatizare prin care beneficiarii să își gestioneze locuințele într-o manieră eficientă și comodă, utilizând dispozitive și tehnologii de ultimă generație. De la sisteme de securitate, sisteme de încălzire sau de iluminat, utilizatorii vor putea controla totul prin diferite interfețe.

Proiectul va fi referit în continuare prin următorul nume de cod: W1S3B33.

Printre obiectivele principale ale proiectului W1S3B33 se numără învățarea unor noi tehnologii, livrarea unei aplicații funcționale care să asigure controlul și monitorizarea echipamente cum sunt cele de iluminat, încălzire, acces sau securitate. Obiectivele finale constau în verificarea și evaluarea performanțelor sistemului dezvoltat astfel încât acesta să se conformeze așteptărilor și cerințelor clientului.

În concluzie, proiectul W1S3B33 demarat de către compania WiseBee este o inițiativă profesională care vizează integrarea unui concept util în mediul cotidian, respectând standardele de siguranță și eficiență impuse de către solicitanți și contribuind astfel la dezvoltarea comunității tehnice din aria de activitate mai sus menționată.

# 2. Testarea codului

Modul Web - partea de cod

În cadrul modulului Web vor exista trei clase controller responsabile de gestionarea fluxului de cereri și răspunsuri de la și către utilizator: AuthenticationController, AdminController și UserController. Codul integral al aplicației web se găsește pe [github](https://github.com/GeoM-bit/Homematic/tree/main/Modul%20Web).

În figura de mai jos se poate observa codul pentru clasa AuthenticationController:

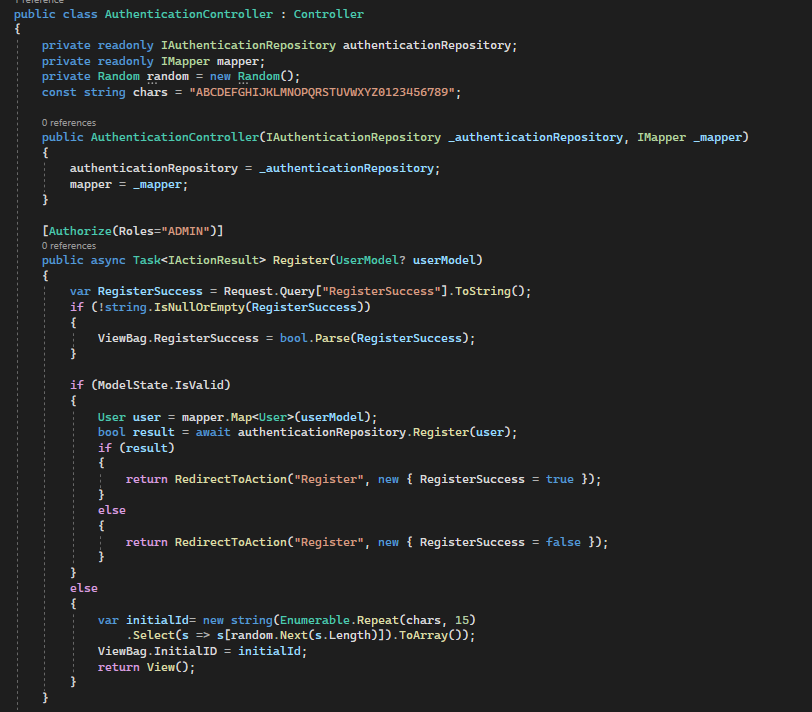




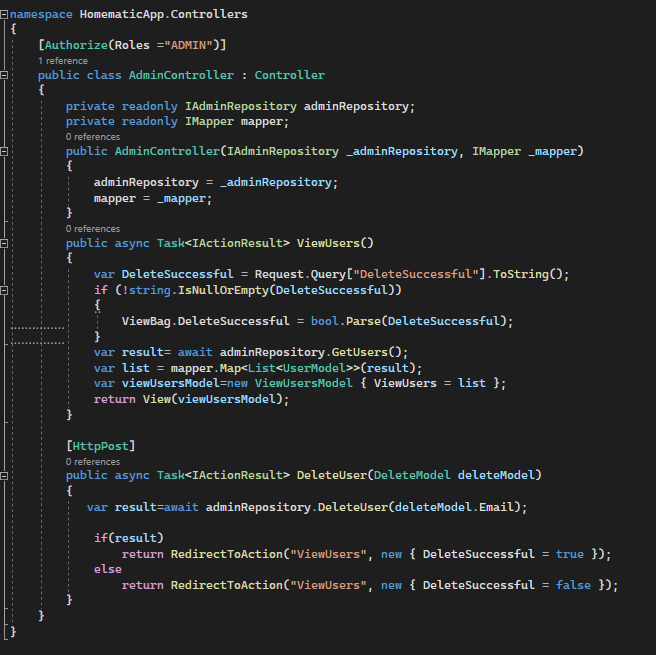
Figura 1. AuthenticationController

Metoda de Register din această clasă va putea fi accesată doar de către administrator, va verifica dacă există erori pe model, iar în cazul în care modelul este valid, se face maparea către un obiect de tip User se va transmite către AuthenticationRepository pentru a fi înregistrat în baza de date. Câmpul Device\_Id va fi populat cu o valoare aleatoare, urmând a fi actualizat cu un cod IMEI valid la prima logare de pe telefonul mobil. Se verifică răspunsul trimis de către repository, iar în funcție de el se redirecționează către aceeași pagină, adăugând în rută indicatorul RegisterSuccess. RegisterSuccess va fi folosit pentru a decide dacă va fi afișată o alertă de înregistrare realizată cu succes sau nu.

Logica de Login a fost împărțită în două metode: metoda LoginUser care se ocupă de primirea datelor din formularul de login, transmiterea lor către repository, setarea tokenului dacă logarea s-a realizat cu succes, și metoda Login care va verifica parametrii din rute, va seta valori în ViewBag ce vor fi folosite pentru afișarea alertelor, va verifica existența unui token, a unui rol și va redirecționa către pagina de ViewParameters.

În Logout, accesibil doar dacă utilizatorul este logat, se va șterge tokenul și se va redirecționa către pagina de Login.

În figura 2 de mai jos este ilustrat codul din AdminController. Administratorul va putea vedea o listă a tuturor utilizatorilor înregistrați, într-un tabel, și va avea posibilitatea să îi șteargă apăsând pe un buton ce va deschide o fereastră de confirmare. De aici se va putea realiza apelul către funcția DeleteUser care va trimite către repository email-ul utilizatorului ce va fi șters



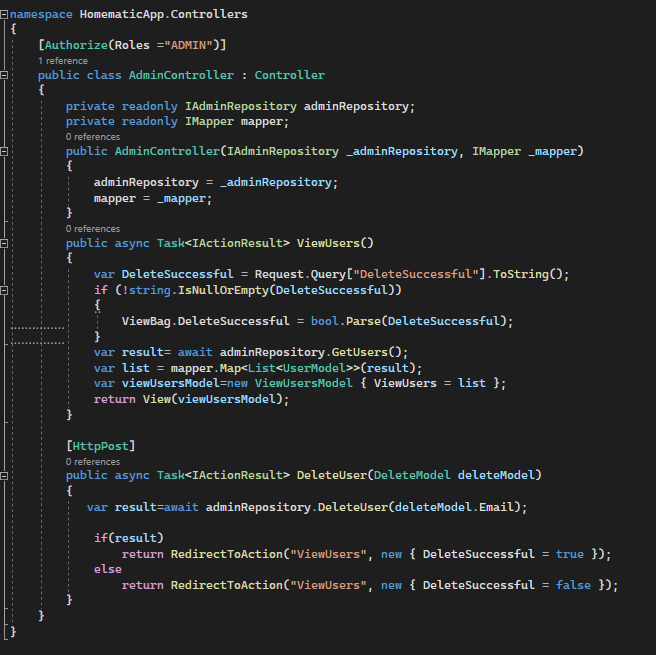
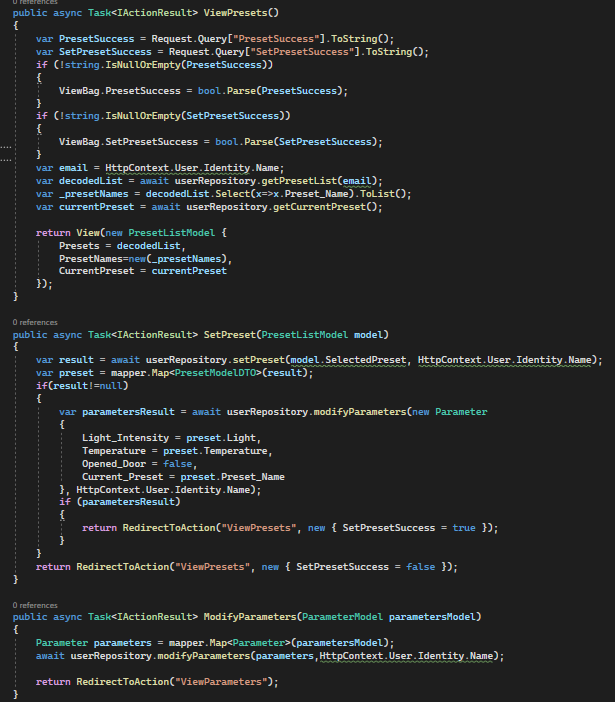


Figura 2. AdminController

Codul aferent clasei de UserController este vizibil în figura 3. Aici se află metodele pentru vizualizarea și modificarea parametrilor, pentru vizualizarea înregistrărilor din tabela de acțiuni și din tabela pentru temperatura esp într-o manieră paginată, pentru vizualizarea, crearea și setarea unui preset și pentru aducerea datelor necesare construirii graficelor utilizând pachetul Chart.js.

Metodele pentru vizualizarea de date funcționează asemănător: atunci când se încarcă pagina corespunzătoare lor, vor fi apelate, se vor verifica rutele pentru parametri care să indice o redirecționare cu succes sau nu, se vor căuta datele disponibile pentru utilizatorul logat cu ajutorul metodelor din UserRepository, se fac verificări dacă s-au returnat valori diferite de null, se fac mapări către modelul din view-ul asociat.





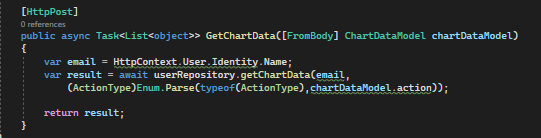
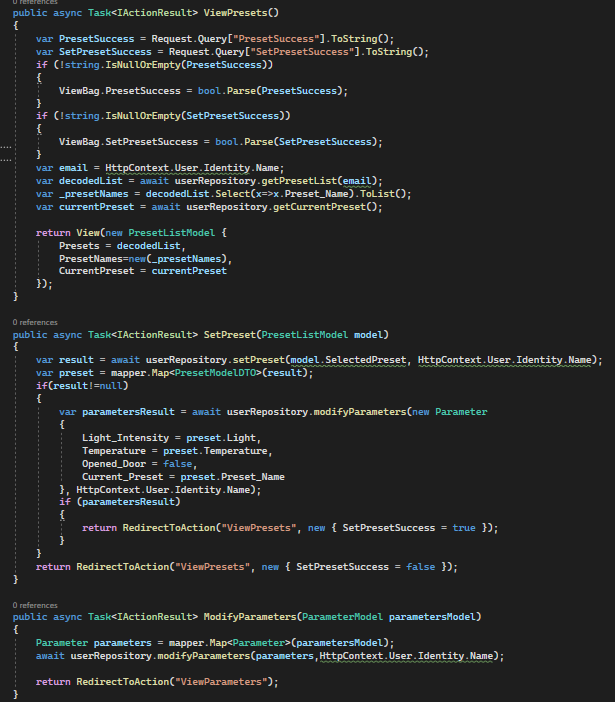


Figura 3. UserController

Modul Web - partea de teste automate

Pentru testarea părții de frontend a aplicației web s-a folosit Cypress (figura 4). S-au testat formularele de login și register, funcționalitatea de ștergere a unui utilizator, apelarea API-ului pentru vizualizarea acțiunilor, pentru vizualizarea și modificarea parametrilor actuali, pentru crearea și setarea unui preset. În figura 5 se poate observa structura folderului de teste și structura unui test în Cypress, iar în figura 6 se pot vedea rezultatele tuturor testelor. Codul testelor în Cypress se găsește de asemenea integral pe [github](https://github.com/GeoM-bit/Homematic/tree/main/Modul%20Web).

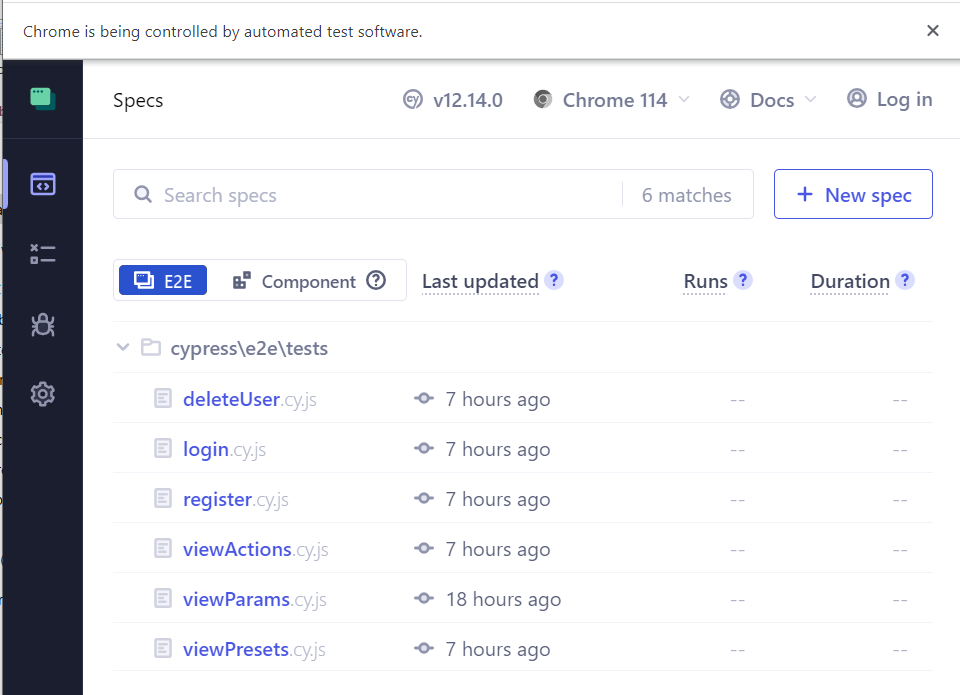


Figura 4. Fișierele de teste din Cypress

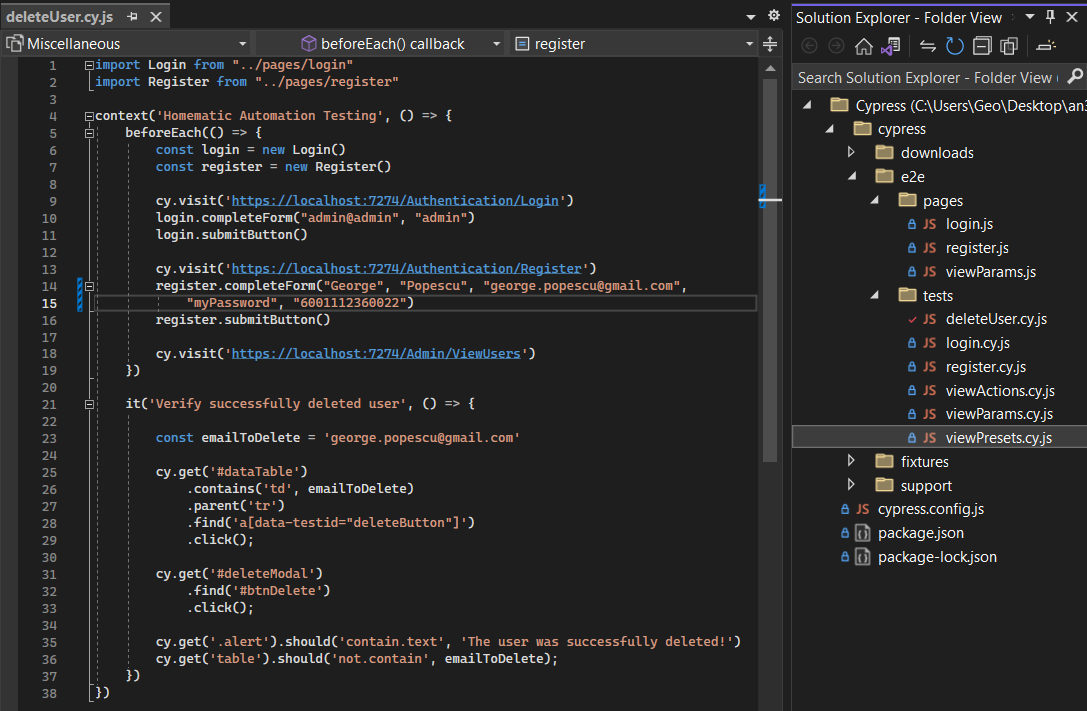
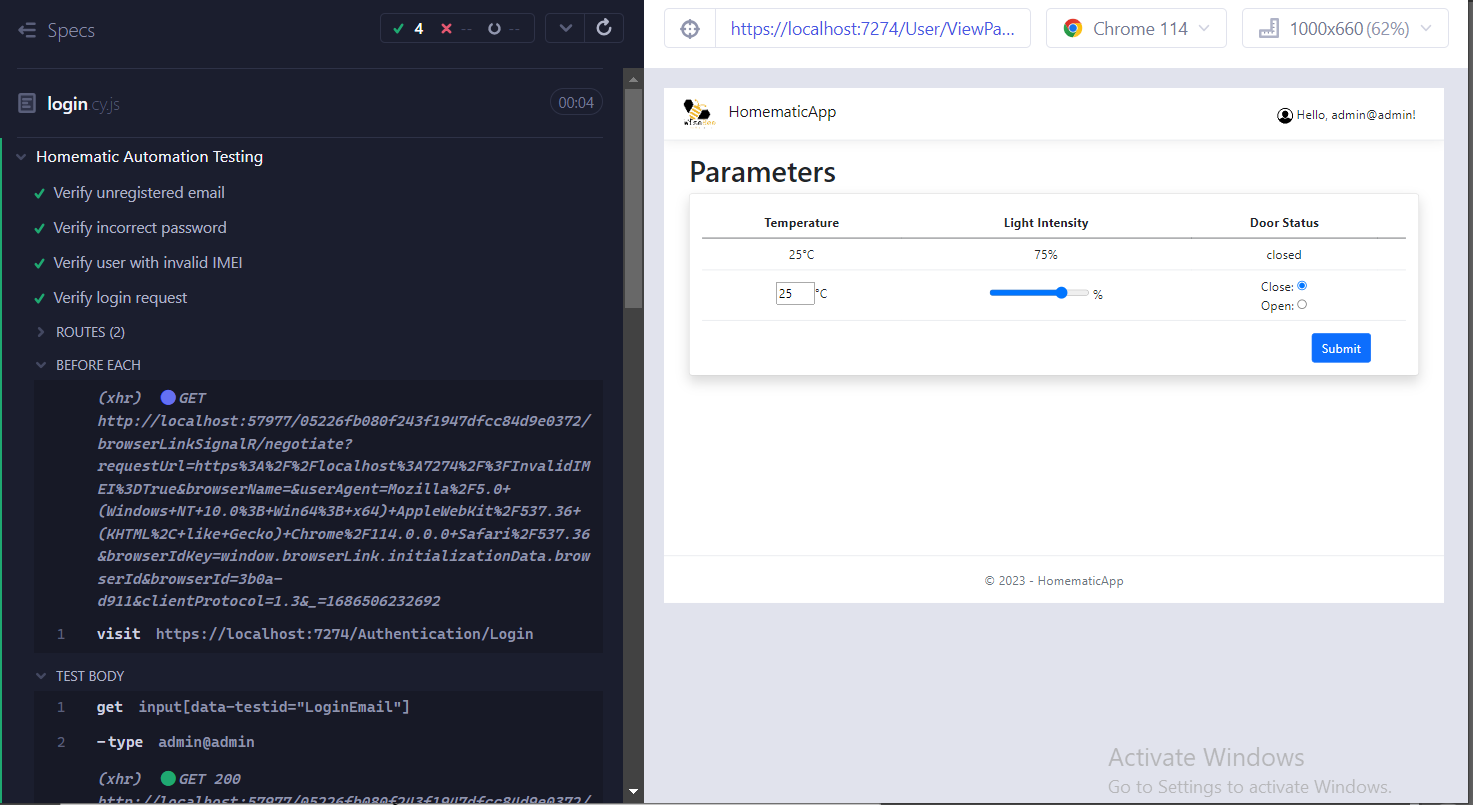
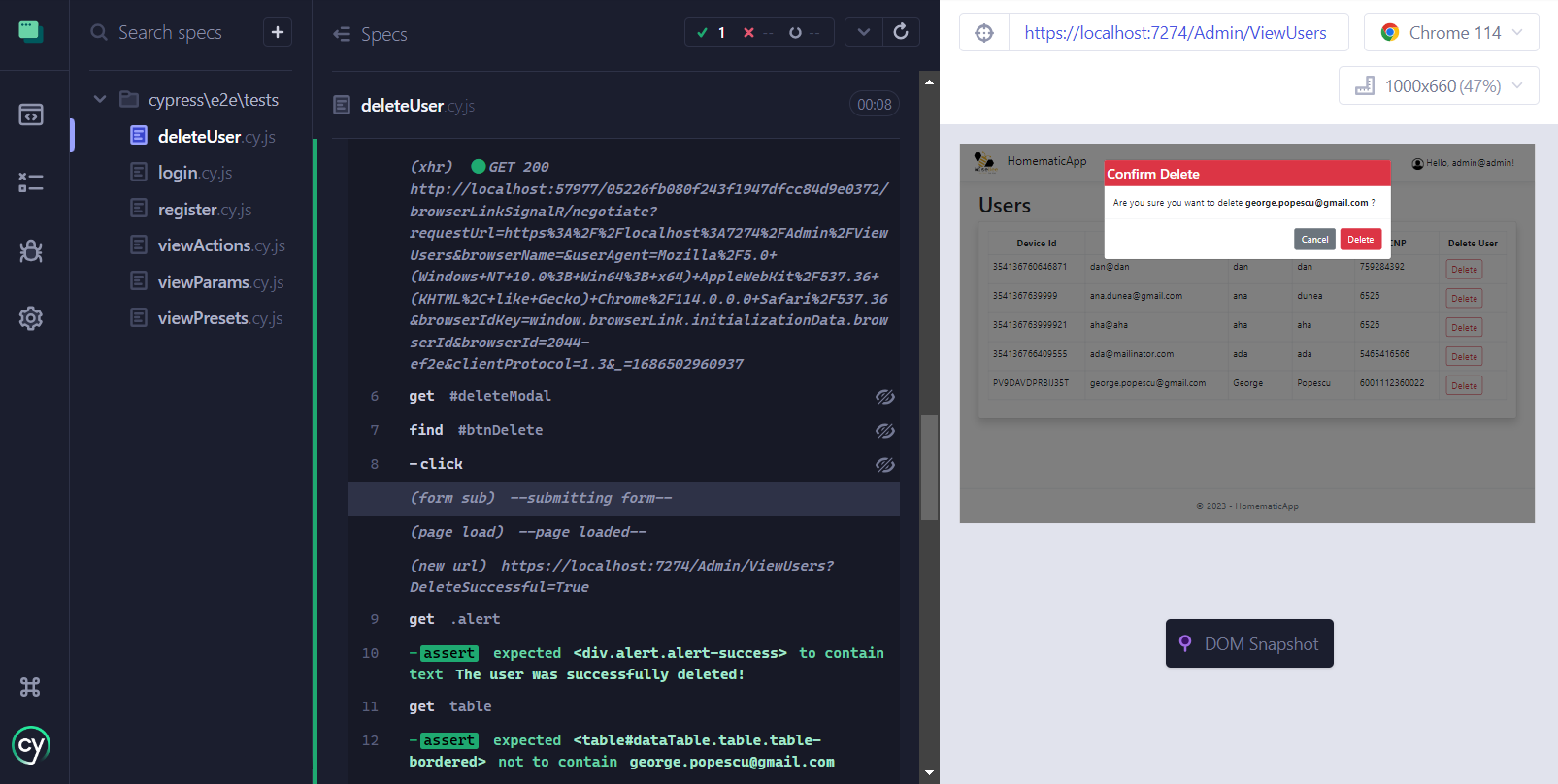
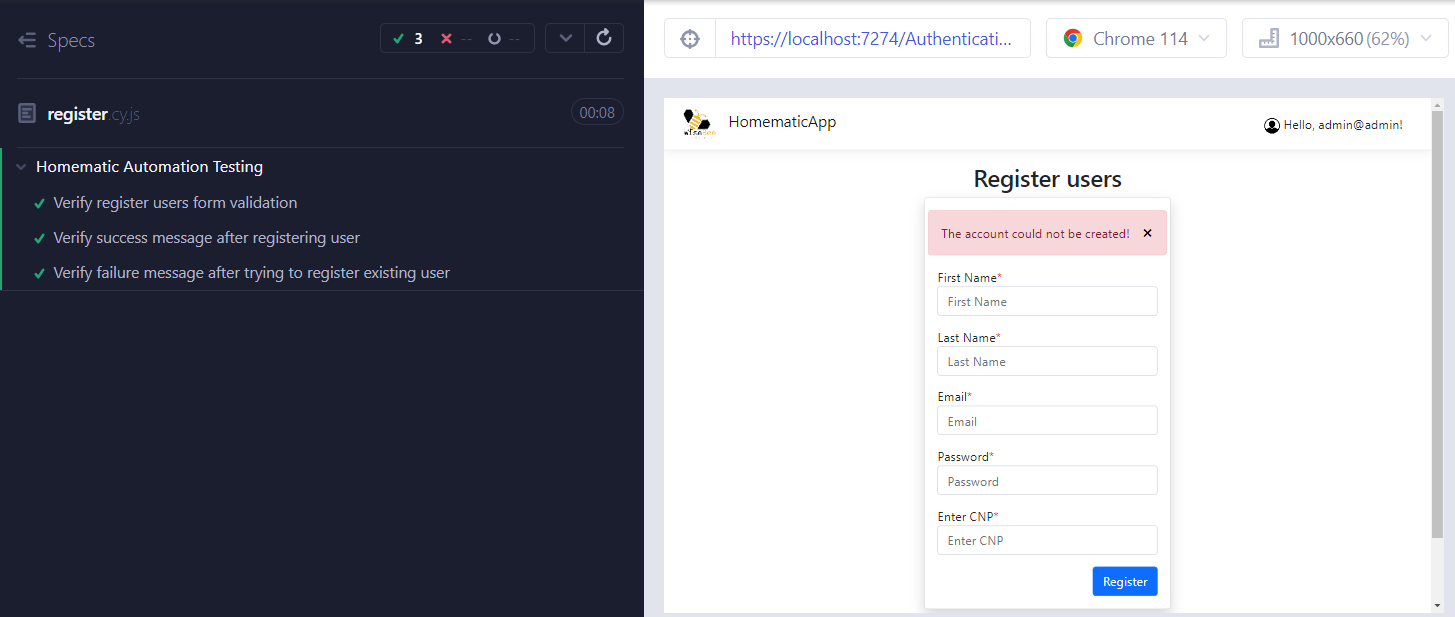
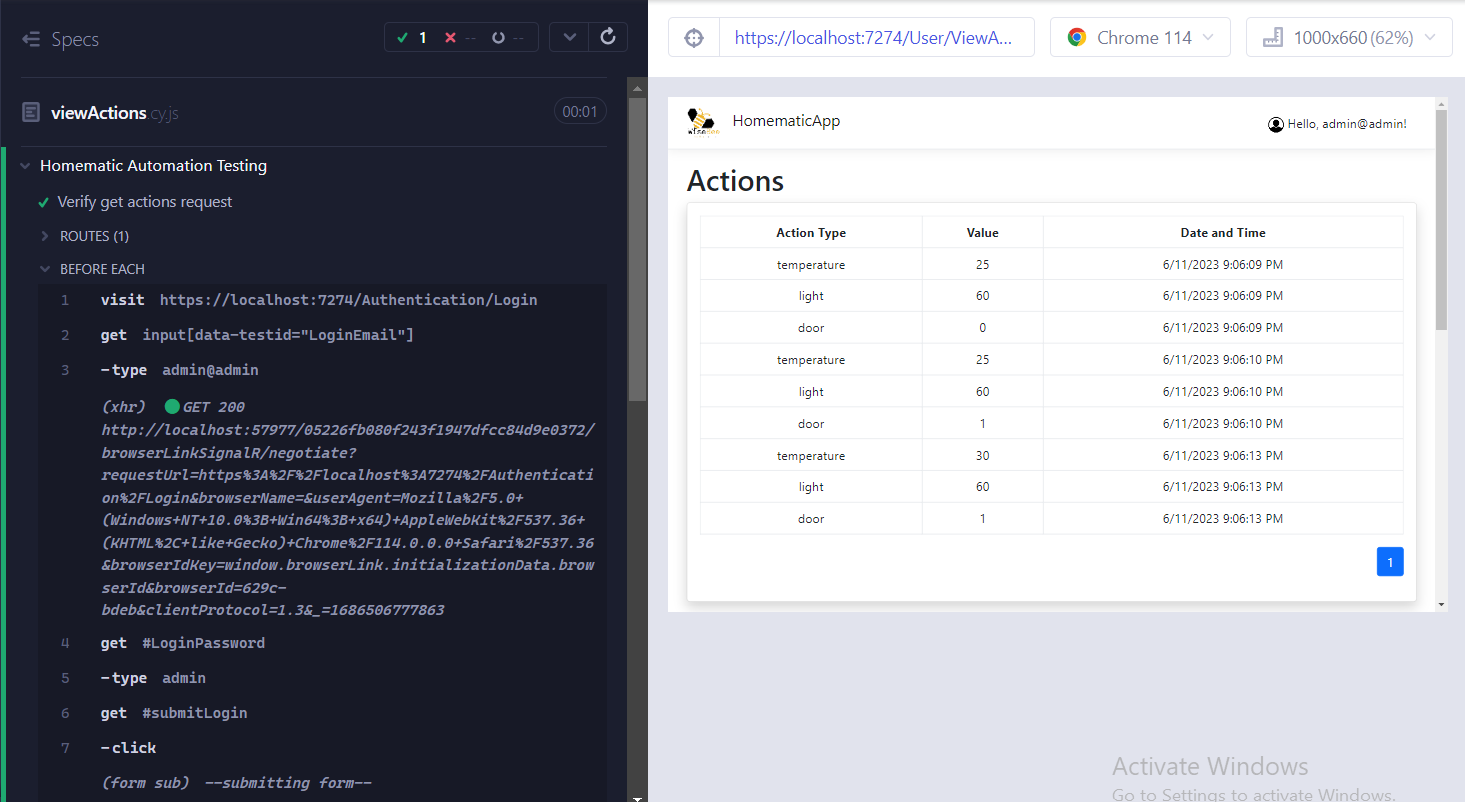


Figura 5. Structura folderului de teste și structura unui test







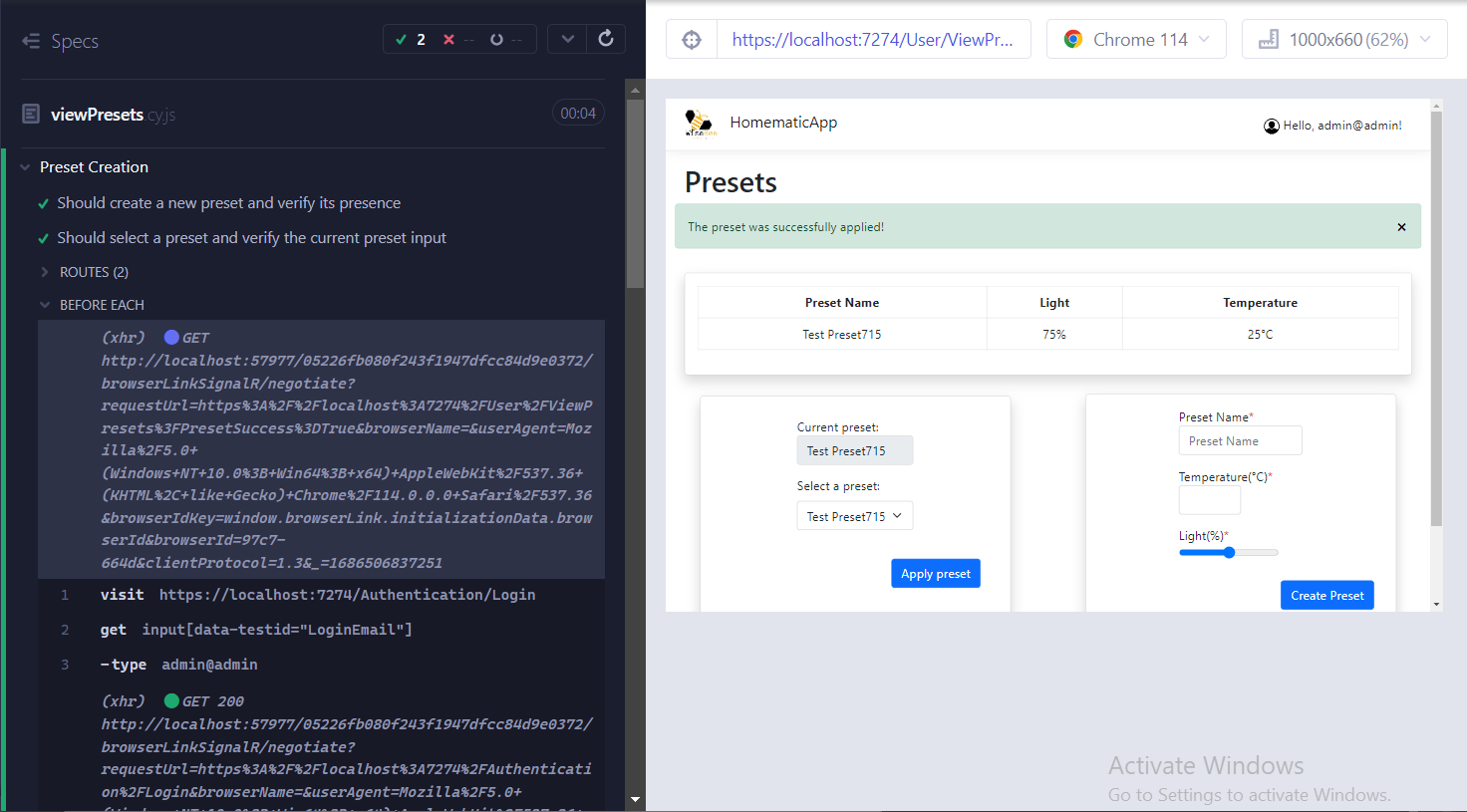
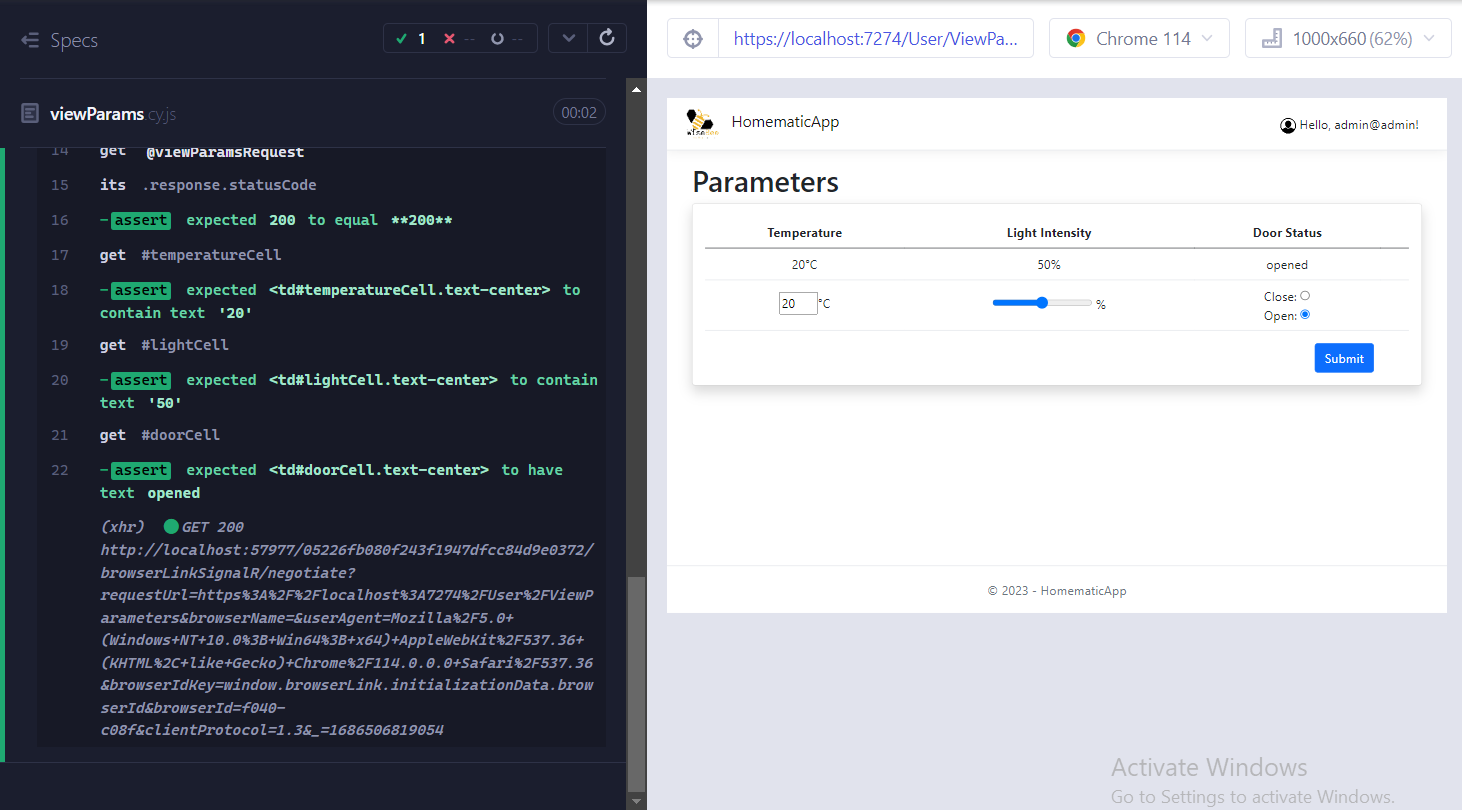
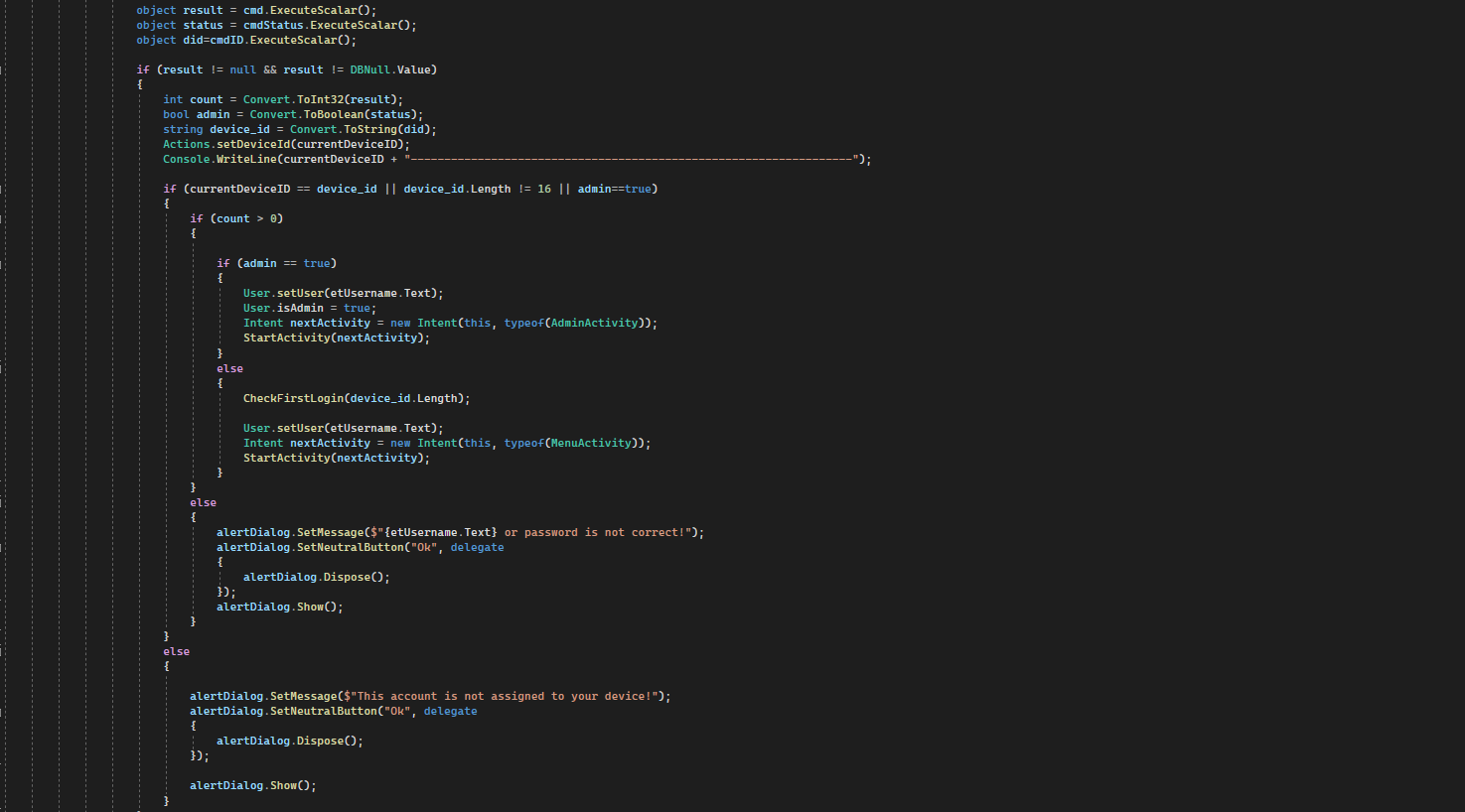
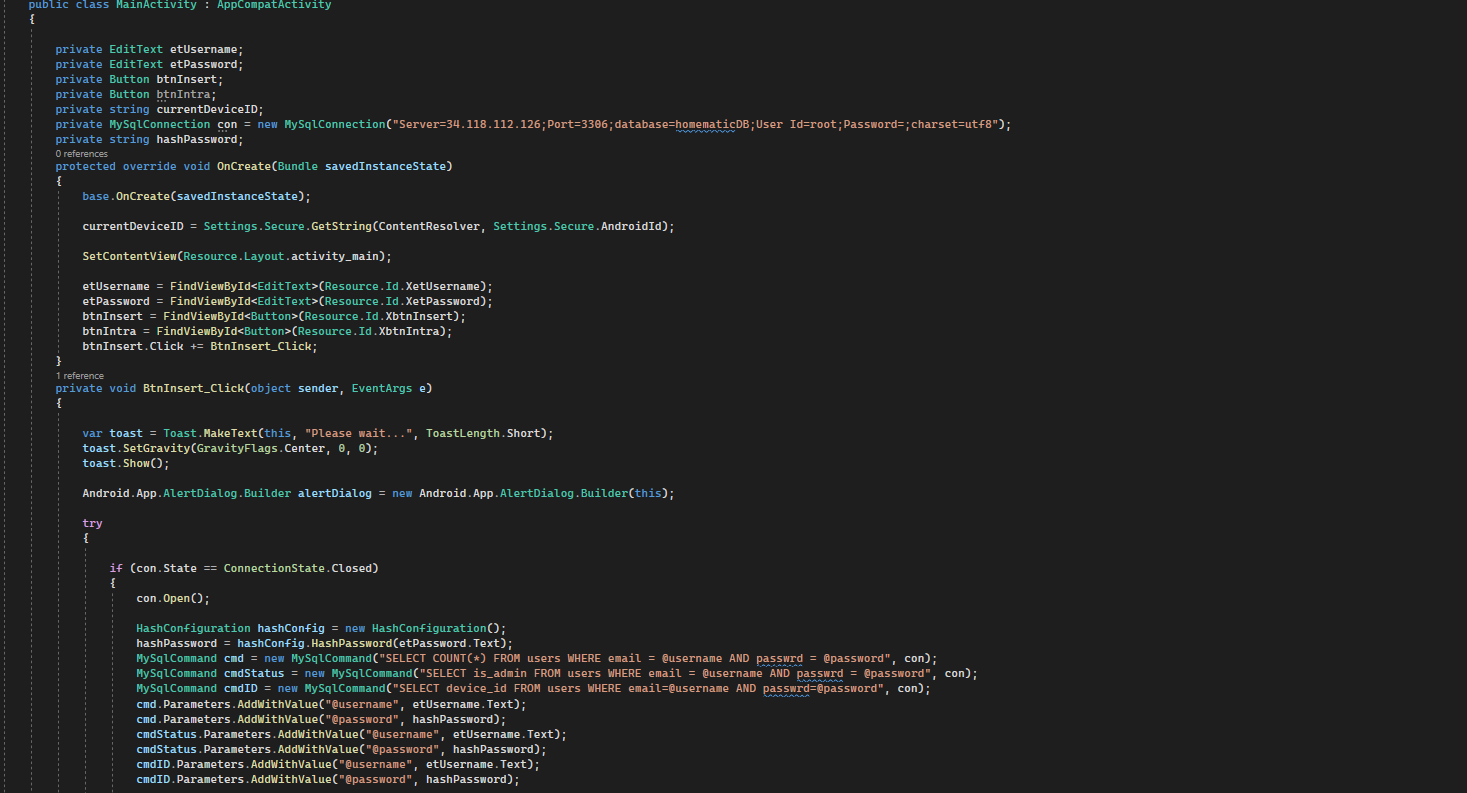


Figura 6. Rezultatele testelor în Cypress

Modul Mobile - partea de cod

Codul complet al aplicației mobile se poate accesa pe [github](https://github.com/AndreiMihai19/XamarinMobileAppProject).

În cazul modulului Mobile clasa care se ocupă de logare și de stabilirea tipului de utilizator (admin sau utilizator) este clasa MainActivity, prezentată mai jos.

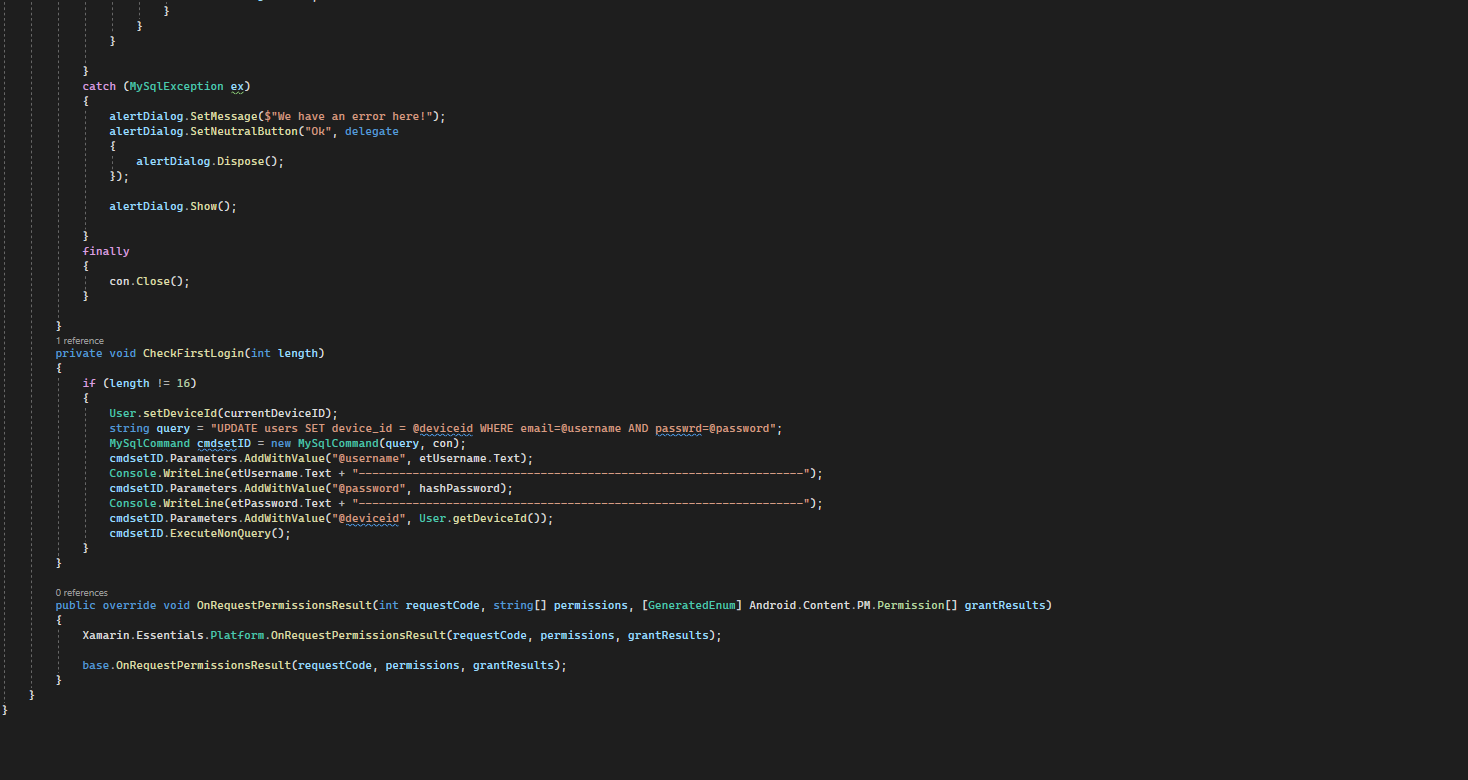


Figura 6. Clasa MainActivity

În cadrul acestei clase se realizează verificare datelor introduse de la tastatură (email-ul și parolă). În cazul în care acestea corespund cu datele stocate în baza de date, se verifică dacă conectarea s-a realizat ca admin sau ca utilizator prin valoare stocată în câmpul is\_admin. Dacă datele corespund, este verificat codul Device\_id, cod unic pentru fiecare dispozitiv android, astfel fiecare utilizator al aplicației poate accesa aplicația de pe un singur dispozitiv, asociat contului sau.

Metoda CheckFirstLogin verifică dacă utilizatorul are asociat un device\_id, în caz contrar acestuia îi este completat automat un string întâmplător la crearea contului de către administrator, iar la prima logare, device\_id-ul dispozitivului este extras și atribuit contului respectiv.

Dacă datele nu corespund cu cele stocate în baza de date sau device\_id-ul este diferit de cel stocat, conectarea nu va putea fi realizată și se va afișa un mesaj corespunzător pe ecran.

În funcție de tipul logari, vom avea următoarele clase care se ocupă de partea de utilizator sau de partea de administrator: MenuActivity și AdminActivity.

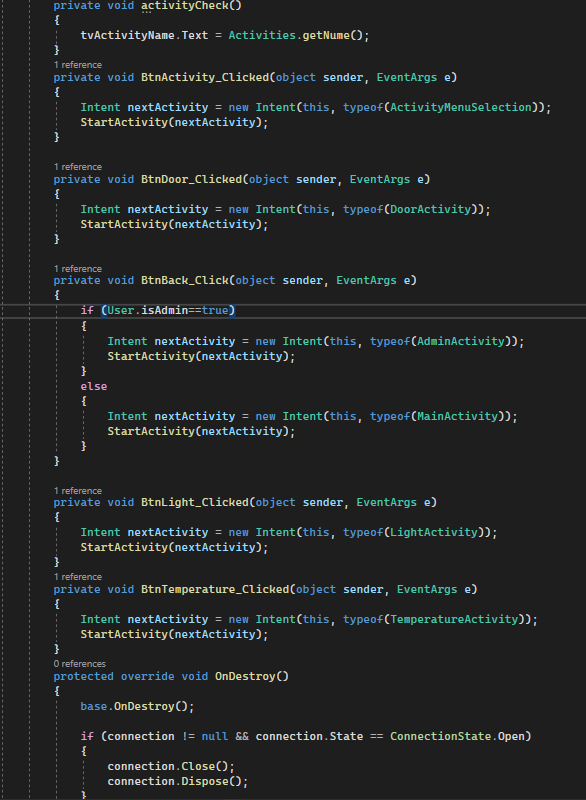
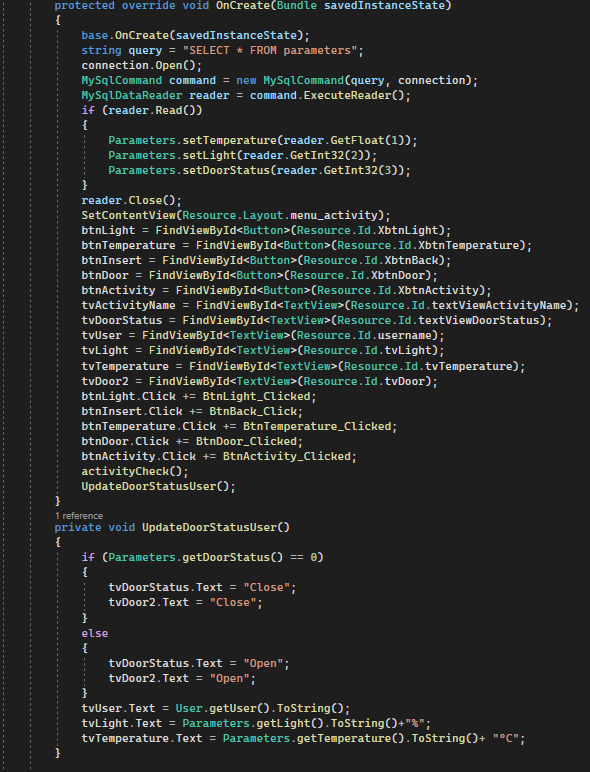


Figura 7. Clasa MenuActivity

Clasa MenuActivity prezentată în Figura 6, are ca rol navigarea prin interfețele aplicației, din perspectiva utilizatorului. Navigarea se face prin intermediul butoanelor btnTemperature, btnDoor, btnLight, btnActivity, care permit accesul la interfețele pentru modificarea individuală a parametrilor, dar și setarea unor activități presetate, sau crearea unor activități noi. (DefaultActivity și PersonalizedActivity).

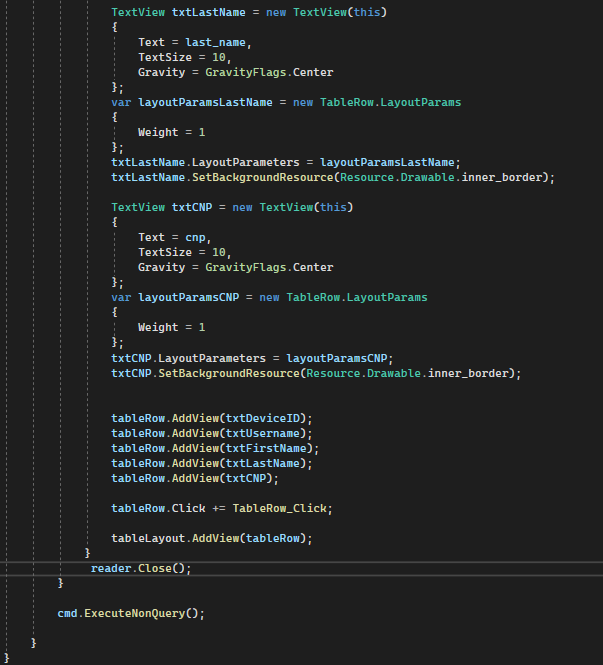
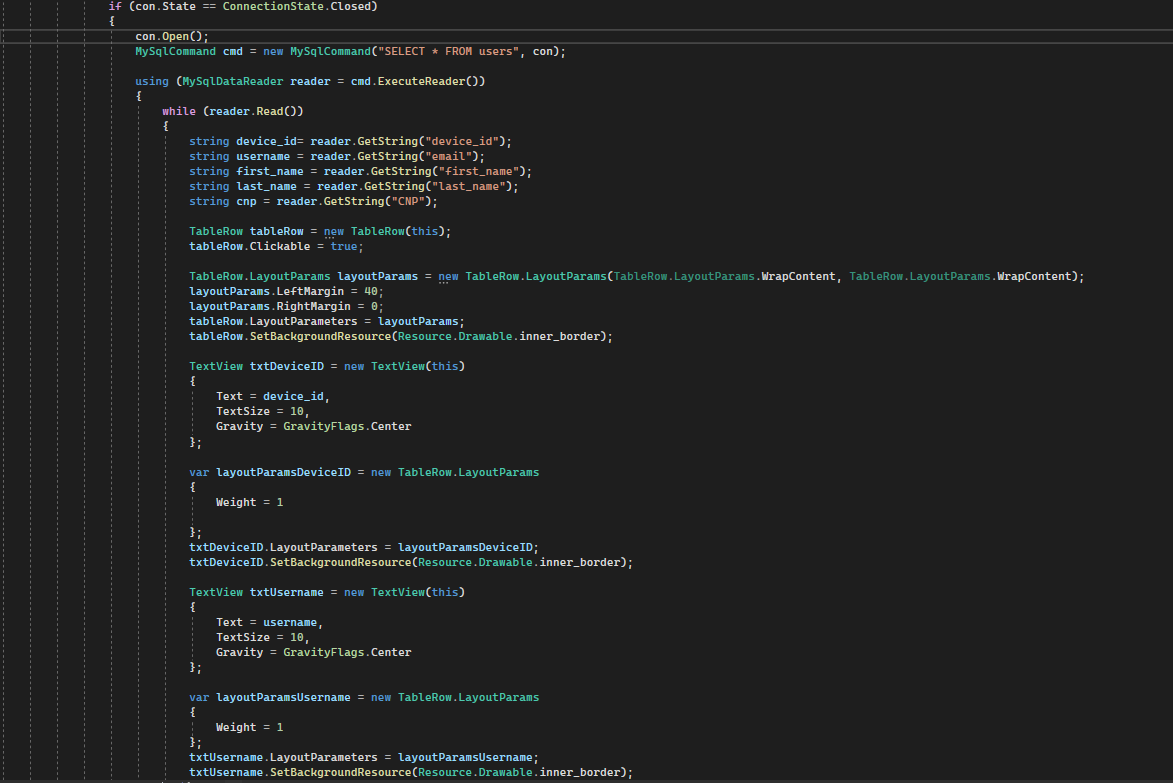
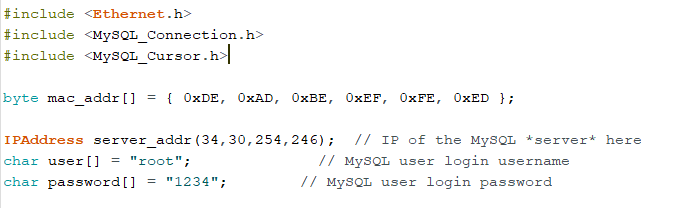


Figura 8. Clasa ShowUsersActivity

Clasa **ShowUsersActivity**, prezentată în figura 8. oferă posibilitatea administratorului de a vizualiza, modifica sau șterge conturile utilizatorilor, prezente în baza de date. Afișarea se face sub forma de tabel în care se regasesc numele, prenumele, device\_id-ul, email-ul și CNP-ul fiecărui utilizator.

Modul Embedded - Arduino Uno  
  
 La nivelul modulului Embedded, pentru partea de Arduino a proiectului, s-a testat conexiunea către modulul Cloud (serverul Google Cloud), cât și preluarea datelor din tabelele bazei de date ( tabela “Parameters” și tabela “Temperature\_ESP”).

Pentru conexiunea către modulul Cloud, s-a testat următorul cod:  


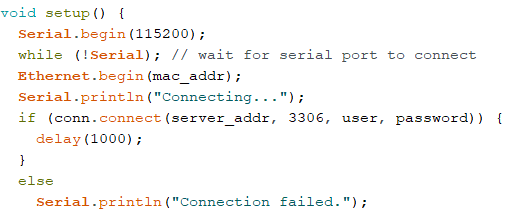
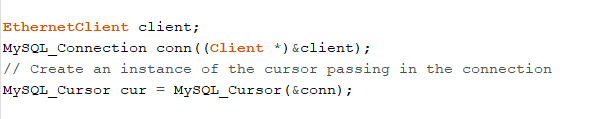


Figura 9. Conectarea la baza de date pentru Arduino

În cazul nostru, variabila server\_addr v-a primi adresa IP a serverului nostru (modulul Cloud), iar datele de autentificare la baza de date sunt atribuite în variabilele “user[]” și “password[]”. Adresa MAC aleasă este cea implicită, aleatoare.  
 După conectarea plăcii Arduino la tensiune, aceasta încearcă să facă conexiunea la baza de date, iar rezultatele s-au putut observa prin mesajele trimise (“Connecting…” sau “Connection failed.”) pe interfața serială către Arduino IDE.

Pentru preluarea datelor din tabelele bazei de date, s-a testat următorul cod:



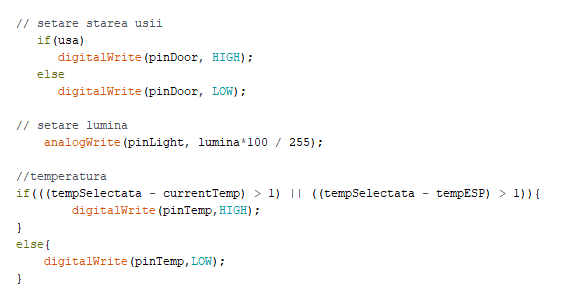
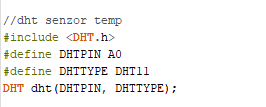


Figura 10. Preluarea datelor din baza de date pentru Arduino

După conectarea plăcii la tensiune, aceasta încearcă să facă conexiunea la baza de date, urmând să parcurgă simultan tabelele “Parameters” și “Temperature\_ESP” și salvând valorile înregistrărilor în variabilele “lumina”, “tempESP”, “tempSelectata” și “usa”. În cazul preluării cu succes a datelor din cele două tabele, se vor întâmpla următoarele: unul dintre led-uri (cel verde) se va aprinde dacă starea ușii este “1”(adică deschis) sau se va stinge dacă starea ușii este “0”(adică închis); ledul albastru va fi aprins folosind tehnica PWM, variabila “lumina” având o valoare naturala din intervalul [0,100]; Ledul roșu se va aprinde dacă încălzirea locuinței este necesară (fie diferența dintre temperatura selectată din aplicația mobile/site-ul web este mai mare cu un grad Celsius față de temperatura de la senzorul de temperatura conectat direct la Arduino, fie diferența dintre temperatura selectată și ultima înregistrare trimisă de la senzorul inteligent de temperatura este mai mare de un grad) și se va stinge dacă condițiile de pornire nu sunt întâlnite.

Pentru preluarea temperaturii de la senzorul direct conectat la Arduino, s-a testat următorul cod:









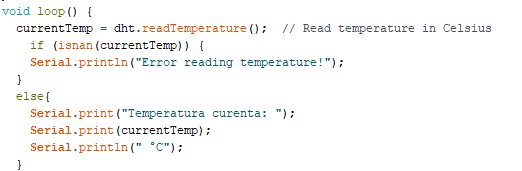
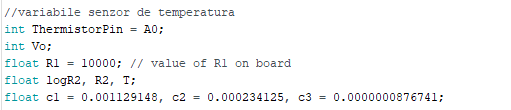


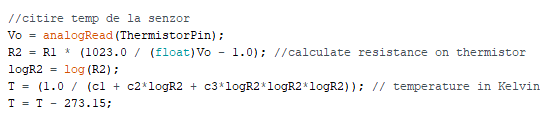
Figura 11. Preluare temperatura senzor Arduino

După conectarea plăcii la tensiune, aceasta pornește în funcția *setup()* senzorul de temperatură, iar în funcția *loop()* se preia temperatura de la senzor(un senzor de temperatura și umiditate DHT11) și se verifică dacă s-a preluat corect. În cazul în care s-a preluat corect, aceasta se trimite pe interfața serială, iar în caz contrar, pe interfața serială se trimite mesajul “Error reading temperature!”.

Modul Embedded - ESP8266  
  
 În cadrul modulului Embedded, pentru partea senzorului ESP8266, s-au testat următoarele: preluarea datelor de la senzorul de temperatura, conectarea la baza de date și inserarea unei înregistrări în tabela “Temperature\_ESP”, funcționarea modului deep sleep.

Pentru testarea preluării datelor de la senzorului de temperatura, s-a testat următorul cod:





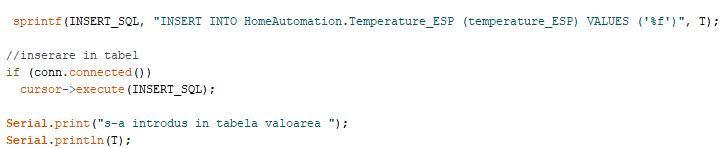
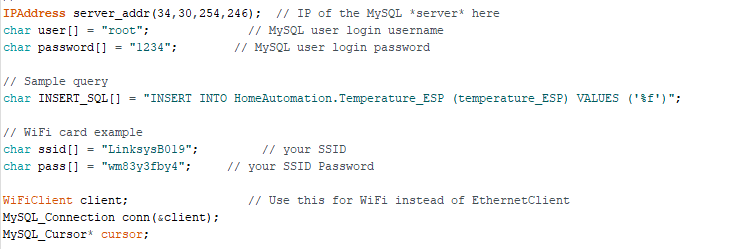
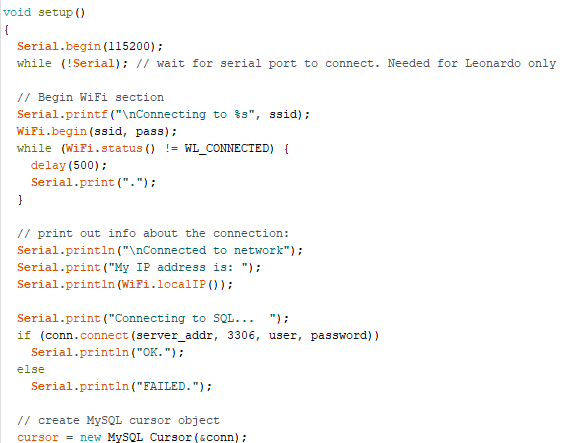


Figura 12. Preluare temperatura senzor ESP8266

După conectarea plăcii ESP8266 la tensiune, aceasta citește datele de la senzorul de temperatură conectat la acesta (un senzor KY-013) și îl introduce în tabela “Temperature\_ESP” a bazei de date. După introducerea noii înregistrări în tabel, temperatura înregistrată de senzor este trimisă și prin interfața serială.

Pentru conectarea la baza de date și inserarea unei noi înregistrări în tabela “Temperature\_ESP”, s-a testat următoarea parte din cod:





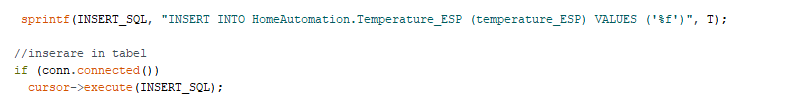


Figura 13. Conectarea la baza de date și inserarea unei înregistrări în tabel

După conectarea plăcii ESP8266 la tensiune, aceasta verifică conexiunea către baza de date, asemănător cu placa Arduino. Dacă conexiunea a avut succes, în tabela ”Temperature\_ESP” se va introduce o nouă înregistrare cu temperatura primită de la senzorul de temperatura. Pentru verificarea faptului ca s-a introdus corect în tabelă, s-au verificat înregistrările tabelei.

Pentru testarea modului *Deep Sleep*, s-a testat următorul cod:  


Figura 14.Modul Deep Sleep

După parcurgerea programului propriu zis, vom seta pin-ul D0 pentru trezirea plăcii ESP din *Deep Sleep* și trimitem comanda de Deep Sleep pentru 60 de secunde. Funcționarea corectă a acestei comenzi poate fi observată la trezirea plăcii, și anume faptul că funcția setup() este chemată din nou, adică se va iar face conectarea la baza de date, preluare temperaturii de la senzor și crearea unei noi înregistrări în tabela “Temperature\_ESP”, înainte de a fi chemată din nou funcția care pornește modul *Deep Sleep.*

Teste de integrare

Pentru verificarea comunicării între module s-au realizat teste de integrare între următoarele module:

* Embedded și Cloud: s-a testat conexiunea cu baza de date pentru citirea, adăugarea, modificarea și înregistrărilor
* Web și Cloud: s-a testat conexiunea cu baza de date pentru citirea, adăugarea, modificarea și ștergerea înregistrărilor
* Mobile și Cloud: s-a testat conexiunea cu baza de date pentru citirea, adăugarea, modificarea și ștergerea înregistrărilor

Teste de sistem

Testele de sistem au fost realizate în scopul validării cerințelor funcționale și nefuncționale. În acest sens, s-au verificat următoarele pentru fiecare modul:

Modul Embedded

1. Comunicarea cu router-ul (și implicit cu cloud-ul) prin Ethernet
2. Comandarea luminilor și a electrovalvelor de calorifer cu ajutorul Arduino
3. Citirea temperaturii de la toți senzorii de temperatură cu ajutorul Arduino
4. Comandarea zăvorului

Modul Web & Mobile

Funcționalități comune pentru toți utilizatorii:

1. Înregistrarea accesului în locuință
2. Vizualizarea datelor parametrilor legate de prezența în casă, temperatură, lumini, starea de închidere a ușii (tabelar și grafic)
3. Solicitarea modificării parametrilor
4. Logarea și delogarea
5. Crearea de evenimente

Funcționalități specifice administratorului:

1. Crearea utilizatorilor pe baza unui formular, aflat pe o pagina dedicată
2. Vizualizarea utilizatorilor înregistrați
3. Ștergerea utilizatorilor prin apăsarea unui buton, aflat într-o coloană adiacentă utilizatorului, cu apariția unei casete de confirmare a ștergerii

Modul Cloud

Parte de stocare

1. Ținerea evidenței utilizatorilor
2. Înregistrarea activității utilizatorilor
3. Înregistrarea evenimentelor
4. Stocarea valorilor pentru buclele de reglare pentru temperatură și lumină
5. Comunicarea cu modulul web, respectiv cu modulul mobile

# 3. Mijloace de verificare

Modul Web

Testarea aplicației web s-a realizat din mediul Visual Studio 2022, pe un browser de Google Chrome, de pe un dispozitiv cu următoarele specificații:

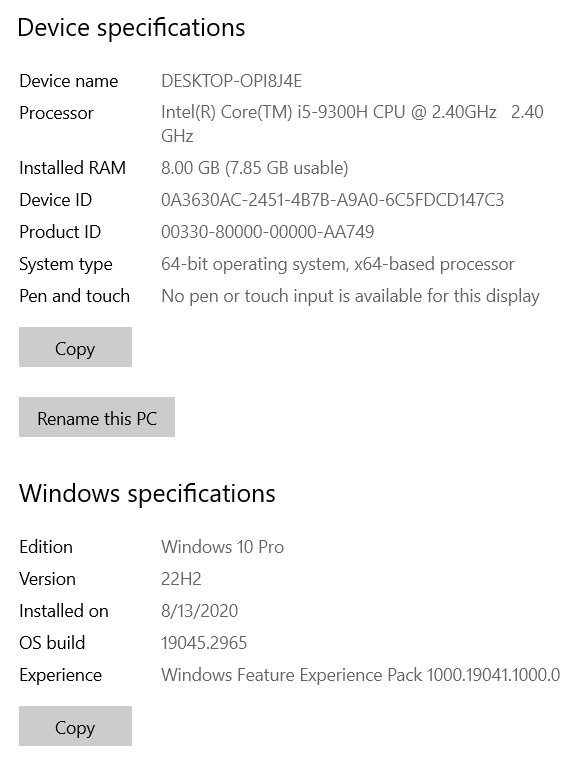


Figura 15. Specificațiile laptopului pe care a fost testată aplicația web

Instrumentul folosit pentru testare a fost Cypress, având următoarea versiune:

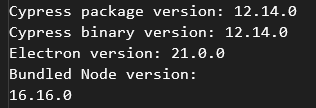


Figura 16. Versiunea Cypress

# 4. Condiții de funcționare

Modul Web

Funcționarea aplicației web necesită conexiunea la internet, pentru a putea face legătura cu baza de date din Google Cloud, existența unui browser în care să poată rula aplicația (Chrome, Edge, Mozilla, etc) - figura 9, un server de pe care să ruleze aplicația(s-a folosit serverul integrat în mediul Visual Studio - figura 10).

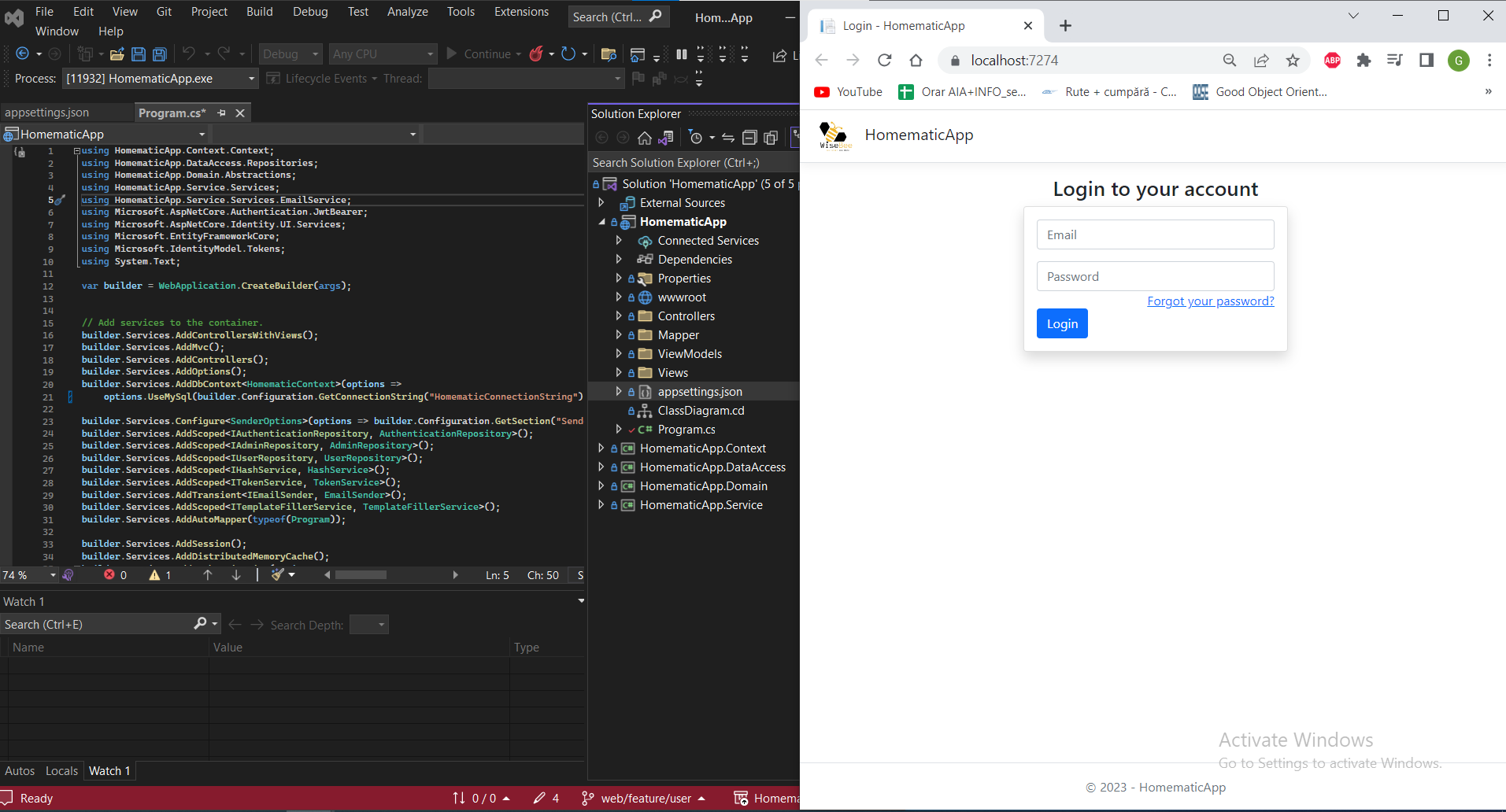


Figura 17. Aplicația rulată într-un browser Google Chrome

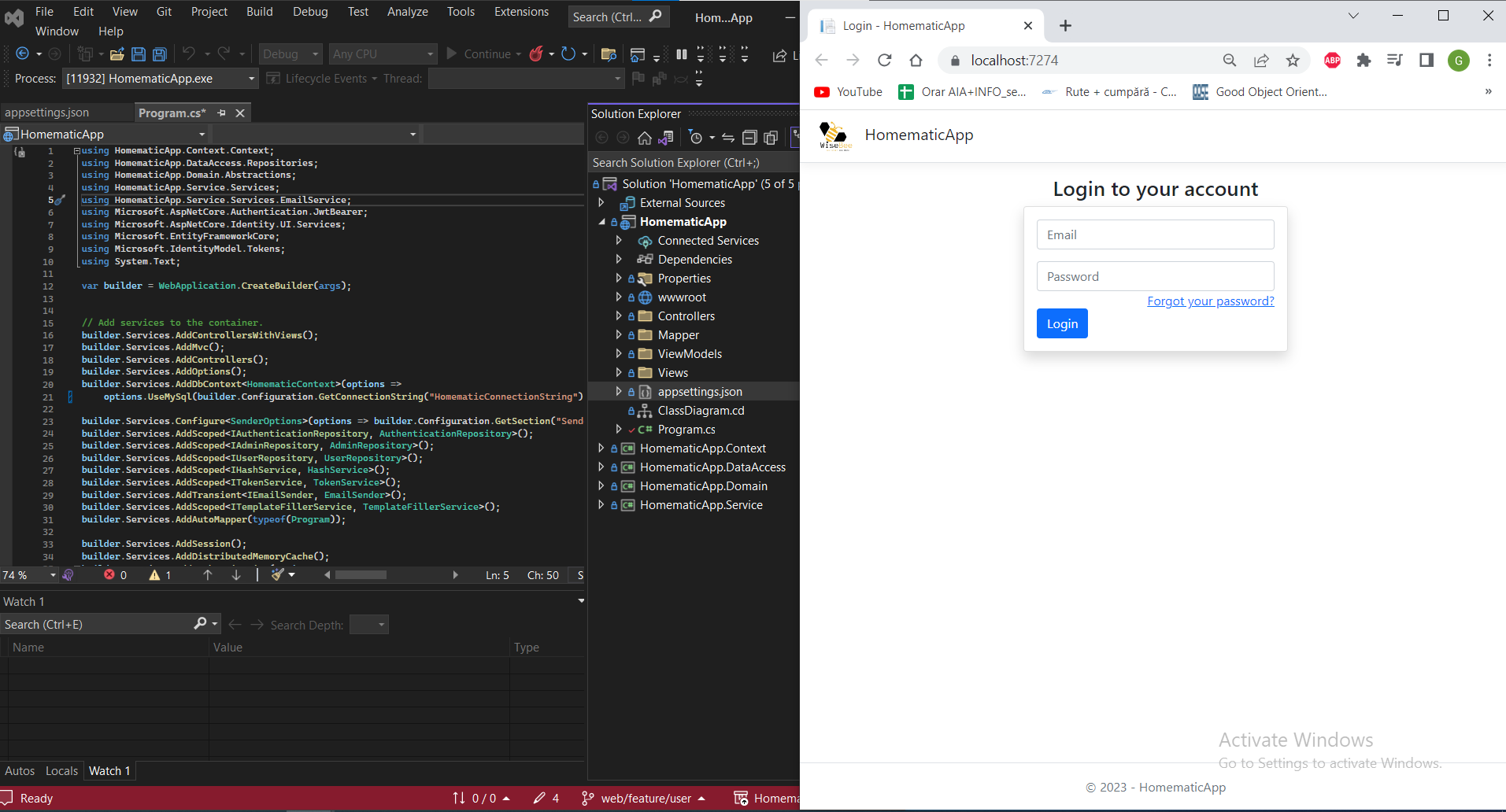


Figura 18. Aplicația rulată din mediul Visual Studio

Modul Mobile

Funcționarea aplicației mobile necesită un dispozitiv mobil Android cu o versiune Android 7.1 sau mai nouă și o conexiunea la internet, pentru a putea face legătura cu baza de date din Google Cloud. Aplicația a fost rulata în mediul **VisualStudio2022.**

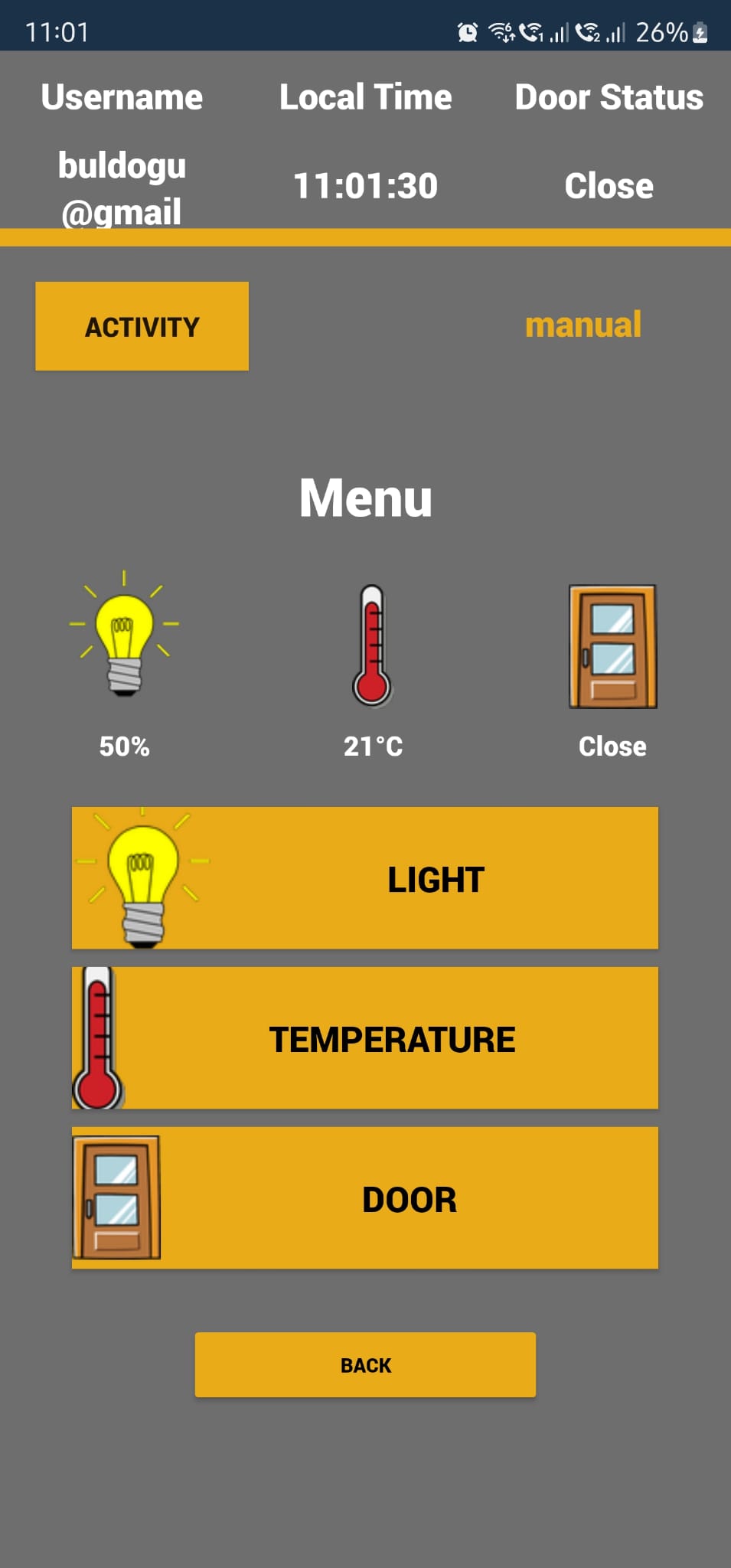
****

Figura 19. Aplicația rulată pe dispozitivul mobil

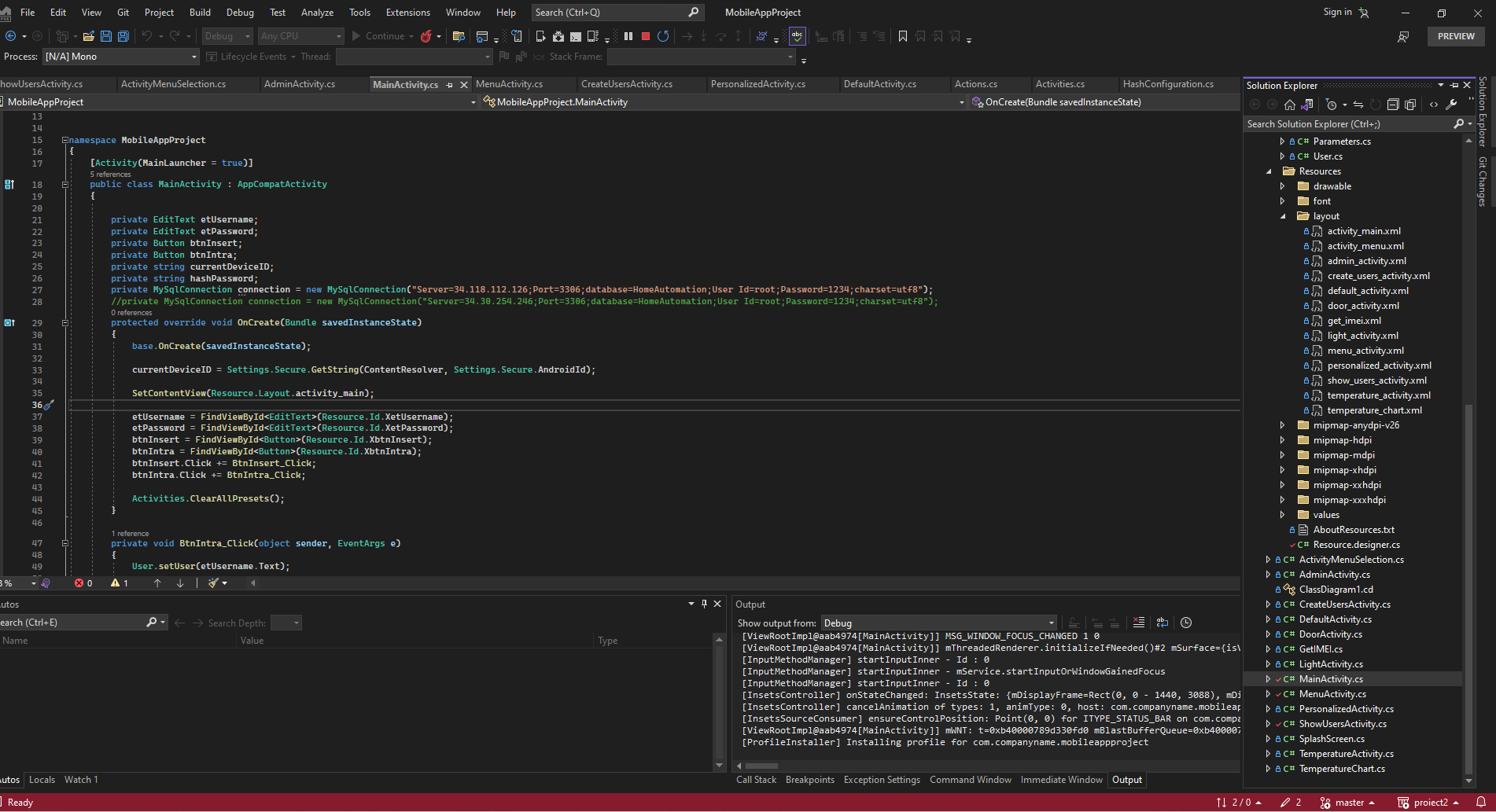


Figura 20. Aplicația mobilă rulată din mediul Visual Studio