



Project Code

Version 1.0

Μέλη της Ομάδας

ΕΠΩΝΥΜΟ	ΟΝΟΜΑ	ΑΡΙΘΜΟΣ ΜΗΤΡΩΟΥ	ΕΤΟΣ
ΚΑΛΥΒΑ	ΚΥΡΙΑΚΗ	1089601	4 ^ο
ΚΥΡΙΑΖΟΠΟΥΛΟΥ	ΒΑΣΙΛΙΚΗ	1090035	4 ^ο
ΜΠΟΤΣΑ	ΠΑΡΑΣΚΕΥΗ	1090058	4 ^ο
ΠΡΟΔΡΟΜΟΥ	ΛΟΥΚΑΣ- ΙΩΑΝΝΗΣ	1084590	4 ^ο

Ευχαριστούμε θερμά για την συμβολή της την φοιτήτριας του 4^{ου} έτους του τμήματος Φαρμακευτικής του Πανεπιστημίου Πατρών Σοφία Ιωαννίδη η οποία μας πρόσφερε ορισμένες ιδέες που θα μπορούσαμε να υλοποιήσουμε την εφαρμογή μας!!!

Editor: Καλύβα Κυριακή

Contributor: Λουκάς Ιωάννης Προδρόμου

Peer Reviewer: Μπότσα Παρασκευή, Κυριαζοπούλου Βασιλική

Περιεχόμενα

Project Code Version 1.0	1
Κώδικας Δημιουργίας βάσης	4
CREATE TABLE Medicine.....	4
CREATE TABLE Citizen(.....	4
CREATE TABLE Doctor(.....	5
CREATE TABLE Pharmacist(.....	5
Περιβάλλον πολίτη	11
Κώδικας:	14
Το κουμπί βοήθεια:	21
Κώδικας:	22
Ορισμός εικόνας εφαρμογής:.....	33
Περιβάλλον Γιατρού:.....	36
Περιβάλλον Φαρμακοποιού:	38

ΣΧΕΤΙΚΑ ΜΕ ΤΗΝ ΥΠΟΛΟΙΨΗ:

Σημείωση 1^η: Αρχικά παραθέτουμε το github στο οποίο λειτουργήσαμε. Στο branch πρόχειρο φαίνεται καλύτερα η όλη εξέλιξη του κώδικά μας ενώ στο branch main δείχνουμε τα τελικά παραδοτέα του κώδικα.

- **Link github:** <https://github.com/vasiakyr/Technologia>

Σημείωση 2^η: Επιπλέον αξίζει να πούμε πως για την υλοποίηση της εφαρμογής μας πέρα ότι επιλέξαμε την αντικειμενοστρεφή γλώσσα προγραμματισμού rython, αποφασίσαμε για την δημιουργία του GUI να αξιοποιήσουμε το PyQt5.

Σημείωση 3^η: Τέλος χωρίσαμε μέρη της εφαρμογής μεταξύ των τεσσάρων ατόμων της ομάδας μας και δεν έχουμε ενώσει ακόμα τους κώδικες, αυτό είναι κάτι που σκοπεύουμε να υλοποιήσουμε στο τελικό παραδοτέο. Οπότε ανάλογα με τα τμήματα της εφαρμογής που έφτιαξε κάθε μέλος παραθέτουμε και τα αντίστοιχα στιγμιότυπα και σχόλια.

ΒΑΣΗ ΔΕΔΟΜΕΝΩΝ

Κώδικας Δημιουργίας βάσης

```
CREATE TABLE Medicine
(
    med_name CHAR(20),
    med_barcode INT(20),
med_cures ENUM('PONOKEFALO', 'PYRETO', 'BHXA' , 'PONOLAIMO' ,
    'ARTHROITIKA' , 'STOMAXOPONO'),
    med_company VARCHAR(200),
    med_isprescription_needed ENUM('YES' , 'NO'),
    PRIMARY KEY(med_barcode)
);
```

```
CREATE TABLE Citizen(
    name VARCHAR(30) NOT NULL DEFAULT 'unknown',
    address VARCHAR(20) NOT NULL DEFAULT 'unknown',
    area VARCHAR(20) NOT NULL DEFAULT 'unknown',
    age INT(3) NOT NULL,
    sex ENUM('MALE', 'FEMALE') NOT NULL,
    AMKA INT(11) NOT NULL,
    username VARCHAR(20) NOT NULL,
    password VARCHAR(20) NOT NULL,
```

```
phone VARCHAR(20) NOT NULL,  
email VARCHAR(40) NOT NULL,  
PRIMARY KEY(cit_AMKA)  
) engine=InnoDB;
```

```
CREATE TABLE Doctor(  
name VARCHAR(30) NOT NULL DEFAULT 'unknown',  
address VARCHAR(20) NOT NULL DEFAULT 'unknown',  
type ENUM('PRIVATE' , 'PUBLIC') NOT NULL,  
specialty ENUM('ODONTIATROS' , 'OFTHALMIATROS' , 'ORTHOPEDIKOS'  
, 'PATHOLOGOS' , 'ENDOKRINOLOGOS' , 'GYNAIKOLOGOS') NOT NULL,  
area VARCHAR(20) NOT NULL DEFAULT 'unknown',  
home_visit ENUM('YES' , 'NO') NOT NULL,  
username VARCHAR(20) NOT NULL,  
password VARCHAR(20) NOT NULL,  
phone VARCHAR(20) NOT NULL,  
email VARCHAR(40) NOT NULL,  
rating INT(2) NULL,  
magic_word ENUM('EYES' , 'BONES' , 'LUNGS') NOT NULL,  
PRIMARY KEY(doc_username)  
) engine=InnoDB;
```

```
CREATE TABLE Pharmacist(  
name VARCHAR(30) NOT NULL DEFAULT 'unknown',  
address VARCHAR(20) NOT NULL DEFAULT 'unknown',  
area VARCHAR(20) NOT NULL DEFAULT 'unknown',  
username VARCHAR(20) NOT NULL,  
timetable TEXT,  
password VARCHAR(20) NOT NULL,  
phone VARCHAR(20) NOT NULL,  
email VARCHAR(40) NOT NULL,  
PRIMARY KEY(pha_username)  
) engine=InnoDB;
```

```
CREATE TABLE Agenda(  
ag_cit_AMKA INT(11) NOT NULL,  
ag_doc_username VARCHAR(20) NOT NULL,  
PRIMARY KEY(ag_cit_AMKA, ag_doc_username),  
CONSTRAINT CITIZENAMKA FOREIGN KEY(ag_cit_AMKA) REFERENCES  
citizen(cit_AMKA)  
ON UPDATE CASCADE ON DELETE CASCADE,  
CONSTRAINT DOCTORUSERNAME FOREIGN KEY(ag_doc_username)  
REFERENCES doctor(doc_username)  
ON UPDATE CASCADE ON DELETE CASCADE  
)engine=InnoDB;
```

```
CREATE TABLE Review(  
rev_cit_AMKA INT(11) NOT NULL,  
rev_doc_username VARCHAR(20) NOT NULL,  
rev_score INT(2) NOT NULL,
```

```

        rev_comments TEXT,
        PRIMARY KEY(rev_cit_AMKA, rev_doc_username),
        CONSTRAINT CITIZEN_AMKA FOREIGN KEY(rev_cit_AMKA)
            REFERENCES citizen(cit_AMKA)
            ON UPDATE CASCADE ON DELETE CASCADE,
        CONSTRAINT DOCTOR_USERNAME FOREIGN KEY(rev_doc_username)
            REFERENCES doctor(doc_username)
            ON UPDATE CASCADE ON DELETE CASCADE
    )engine=InnoDB;

```

```

CREATE TABLE BloodDonation(
    bld_doc_username VARCHAR(20) NOT NULL,
    bld_address VARCHAR(20) NOT NULL DEFAULT 'unknown',
    bld_program TEXT,
    PRIMARY KEY (bld_doc_username),
    CONSTRAINT doctor_at FOREIGN KEY (bld_doc_username)
    REFERENCES doctor(doc_username) ON UPDATE CASCADE ON DELETE
    CASCADE
)ENGINE=InnoDB;

```

```

CREATE TABLE Pending_Reservations(
    pr_customer_name VARCHAR(20) NOT NULL DEFAULT 'unknown',
    pr_number INT(20) NOT NULL,
    pr_med_barcode INT(20),
    PRIMARY KEY (pr_med_barcode),
    CONSTRAINT MEDCODE FOREIGN KEY (pr_med_barcode)
    REFERENCES medicine(med_barcode) ON UPDATE CASCADE ON
    DELETE CASCADE
)ENGINE=InnoDB;

```

```

CREATE TABLE Storage(
    st_pha_name_surname VARCHAR(20) NOT NULL DEFAULT 'unknown',
    st_quantity INT(20) NOT NULL,
    st_med_barcode INT(20),
    PRIMARY KEY (st_med_barcode),
    CONSTRAINT STMEDCODE FOREIGN KEY (st_med_barcode)
    REFERENCES medicine(med_barcode) ON UPDATE CASCADE ON
    DELETE CASCADE
)ENGINE=InnoDB;

```

```

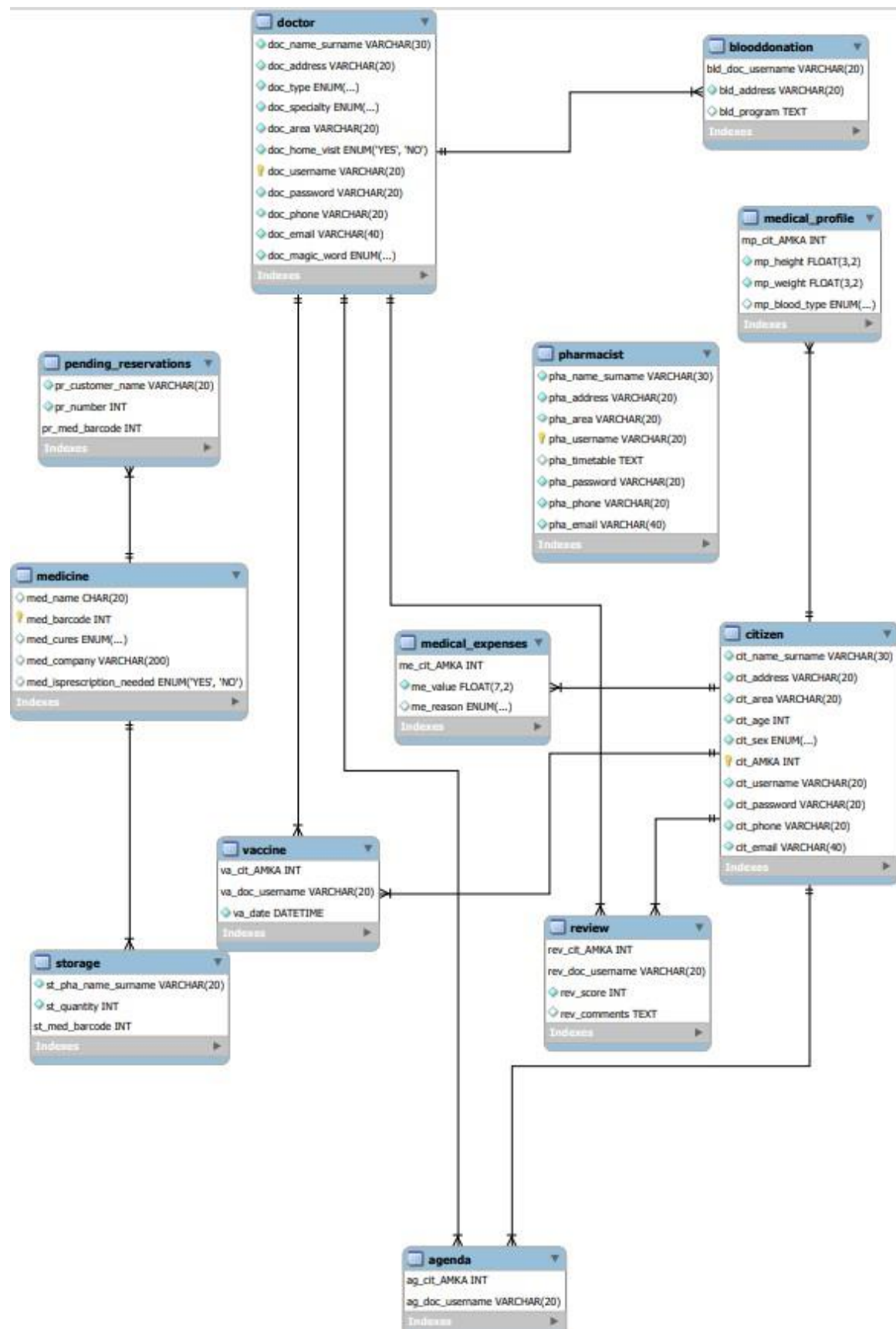
CREATE TABLE Medical_Profile(
    mp_cit_AMKA INT(11) NOT NULL,
    mp_height FLOAT(3,2) NOT NULL,
    mp_weight FLOAT(3,2) NOT NULL,
    mp_blood_type ENUM('O-', 'O+', 'B-', 'B+', 'A-', 'A+', 'AB-', 'AB+'),
    PRIMARY KEY (mp_cit_AMKA),
    CONSTRAINT AMKA FOREIGN KEY (mp_cit_AMKA) REFERENCES
    citizen(cit_AMKA) ON UPDATE CASCADE ON DELETE CASCADE
)ENGINE=InnoDB;

```

```
CREATE TABLE Vaccine(  
    va_cit_AMKA INT(11) NOT NULL,  
    va_doc_username VARCHAR(20) NOT NULL,  
    va_date DATETIME NOT NULL,  
    PRIMARY KEY (va_cit_AMKA , va_doc_username),  
    CONSTRAINT VA_AMKA FOREIGN KEY (va_cit_AMKA) REFERENCES  
        citizen(cit_AMKA) ON UPDATE CASCADE ON DELETE CASCADE,  
    CONSTRAINT VA_USERNAME FOREIGN KEY (va_doc_username)  
        REFERENCES doctor(doc_username) ON UPDATE CASCADE ON DELETE  
        CASCADE  
    )ENGINE=InnoDB;
```

```
CREATE TABLE Medical_Expenses(  
    me_cit_AMKA INT(11) NOT NULL,  
    me_value FLOAT(7,2) NOT NULL,  
    me_reason ENUM('Medicines' , 'Appointment'),  
    PRIMARY KEY (me_cit_AMKA),  
    CONSTRAINT ME_AMKA FOREIGN KEY (me_cit_AMKA) REFERENCES  
        citizen(cit_AMKA) ON UPDATE CASCADE ON DELETE CASCADE  
    )ENGINE=InnoDB;
```

ΔΙΑΓΡΑΜΜΑ ΒΑΣΗΣ ΠΟΥ ΠΡΟΚΥΠΤΕΙ:



Σύνδεση βάσης με εφαρμογή

```
import mysql.connector
```

Εισάγεται αυτή η εφαρμογή για σύνδεση και επικοινωνία με την βάση δεδομένων της MySQL.

```
def create_home_page(self, user_data):
    home_widget = QWidget()
    home_layout = QVBoxLayout()

    greeting_label = QLabel(f'Welcome, {user_data["name"]}!', self)
    greeting_label.setAlignment(Qt.AlignCenter)
    greeting_label.setStyleSheet("font-size: 20px; font-weight: bold;")
    home_layout.addWidget(greeting_label)

    bottom_layout = QHBoxLayout()

    left_box = QVBoxLayout()
    left_label = QLabel('Προσωπικά Στοιχεία', self)
    left_label.setAlignment(Qt.AlignCenter)
    left_label.setStyleSheet("font-size: 16px; font-weight: bold;")
    left_box.addWidget(left_label)

    self.name_label = QLabel('Όνοματεπώνυμο:', self)
    self.name_edit = QLineEdit(self)
    self.name_edit.setReadOnly(True)
    self.name_edit.setText(user_data["name"])
    left_box.addWidget(self.name_label)
    left_box.addWidget(self.name_edit)

    self.email_label = QLabel('email:', self)
    self.email_edit = QLineEdit(self)
    self.email_edit.setReadOnly(True)
    self.email_edit.setText(user_data["email"])
    left_box.addWidget(self.email_label)
    left_box.addWidget(self.email_edit)

    self.phone_label = QLabel('Τηλέφωνο:', self)
    self.phone_edit = QLineEdit(self)
    self.phone_edit.setReadOnly(True)
    self.phone_edit.setText(user_data["phone"])
    left_box.addWidget(self.phone_label)

    left_box.addWidget(self.phone_edit)
```

Όταν συνδέεται ο χρήστης, εμφανίζεται στην αρχική ένα μήνυμα χαιρετισμού (το σύστημα παίρνει από την βάση το όνομα του χρήστη). Επίσης το σύστημα παίρνει από την βάση και άλλα στοιχεία του χρήστη όπως τηλέφωνο και email και τα εμφανίζει στα αντίστοιχα πεδία:

Παράδειγμα:

```
def logout(self):
    self.close() # Close the main window
    self.login_window = LoginWindow() # Show the login window
    self.login_window.show()
```

Όταν ο χρήστης πατάει το κουμπί «Έξοδος» το σύστημα τον ανακατευθύνει στο Login Window:

```
def validate_user(self, username, password):
    try:
        # Establish connection to MySQL
        conn = mysql.connector.connect(
            host="localhost",
            user="root",
            password="1084590",
```

```

        database="medlink"
    )
    cursor = conn.cursor(dictionary=True) # Use dictionary cursor to fetch data as a
dictionary

    tables = ['citizen', 'doctor', 'pharmacist'] # Replace with your actual table names

    for table in tables:
        cursor.execute(f"SELECT * FROM {table} WHERE username=%s AND
password=%s", (username, password))
        result = cursor.fetchone()
        if result:
            conn.close()
            return result

    conn.close()
    return None
except mysql.connector.Error as err:
    print("Error:", err) # Handle connection errors
    QMessageBox.warning(self, 'Database Error', f"An error occurred: {err}")
    return None

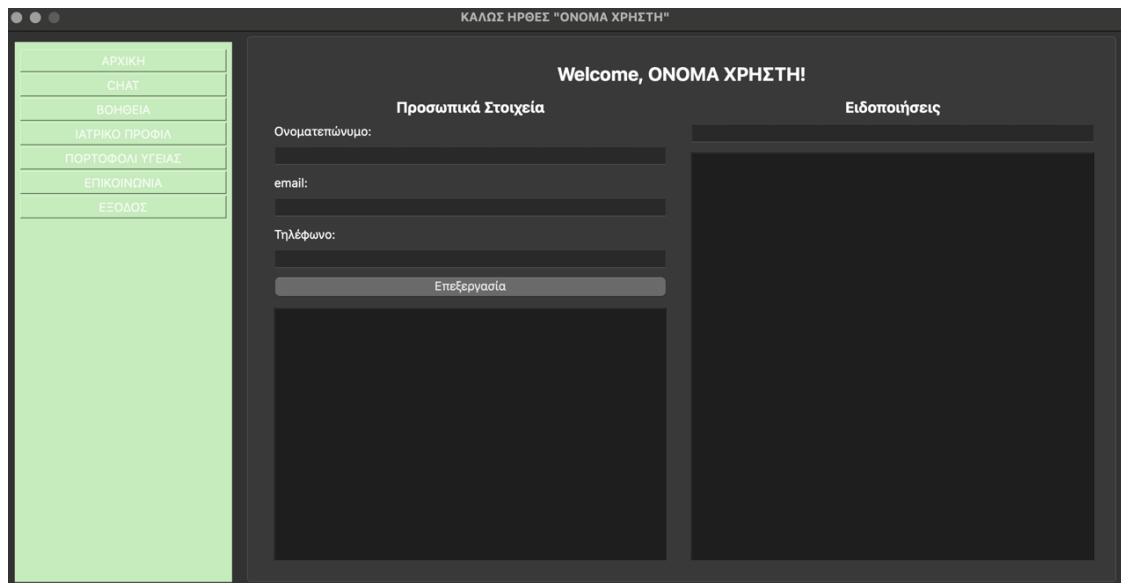
def accept_login(self, user_data):
    self.main_window = MainWindow(user_data)
    self.main_window.show()
    self.close()

```

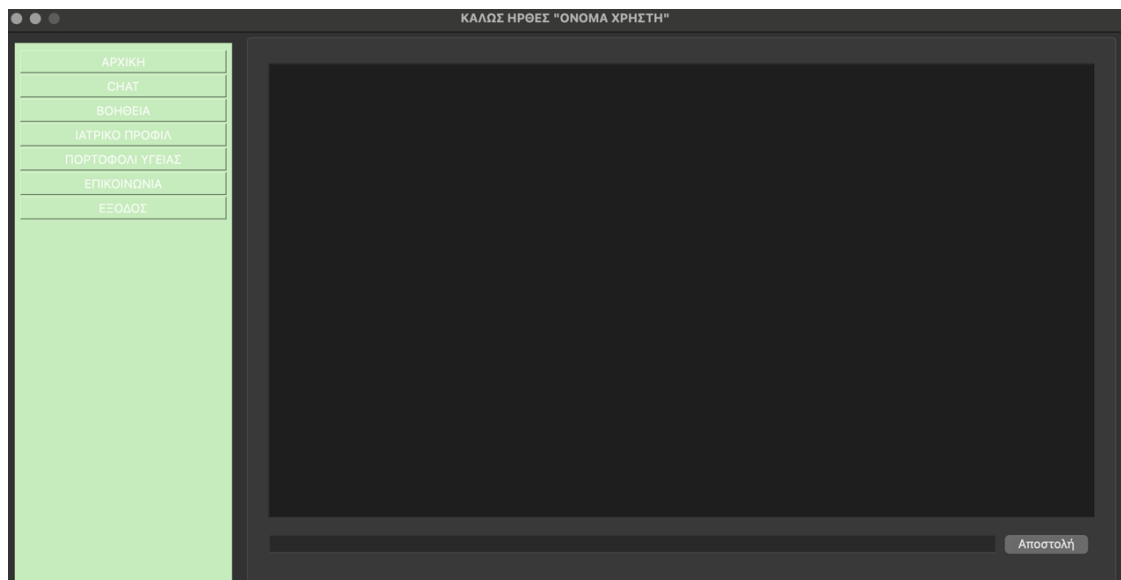
Πραγματοποιούμε σύνδεση με την Βάση που έχουμε δημιουργήσει δίνοντας τα στοιχεία που ζητούνται. Το σύστημα ελέγχει τους πίνακες citizen, doctor, pharmacist για να ελέγξει αν υπάρχει ο χρήστης που προσπαθεί να κάνει Login. Αν υπάρχει τον ανακατευθύνει στο Main Window.

Περιβάλλον πολίτη

Εδώ φαίνεται πως θα βλέπει ο πολίτης την εφαρμογή μας όταν εισέρχεται σε αυτήν και τη χρησιμοποιεί:



Στην αρχή, ο πολίτης μπαίνει στην αρχική οθόνη.
Στην συνέχεια, μόλις πατήσει το κουμπί CHAT θα του ανοίξει την σελίδα επικοινωνίας και έπειτα θα μπορεί να επικοινωνήσει μέσω chat με κάποιο γιατρό:



Με το κουμπί ΙΑΤΡΙΚΟ ΠΡΟΦΙΛ θα μπορεί ο πολίτης να δει τα στοιχεία που έχει καταχωρήσει στην εφαρμογή και να επεξεργαστεί όποια του επιτρέπονται:

ΚΑΛΩΣ ΗΡΘΕΣ "ΟΝΟΜΑ ΧΡΗΣΤΗ"

ΑΡΧΙΚΗ

CHAT

ΒΟΗΘΕΙΑ

ΙΑΤΡΙΚΟ ΠΡΟΦΙΛ

ΠΟΡΤΟΦΟΛΙ ΥΓΕΙΑΣ

ΕΠΙΚΟΙΝΩΝΙΑ

ΕΞΟΔΟΣ

Όνομα:

Επώνυμο:

Ηλικία:

Βάρος:

Ύψος:

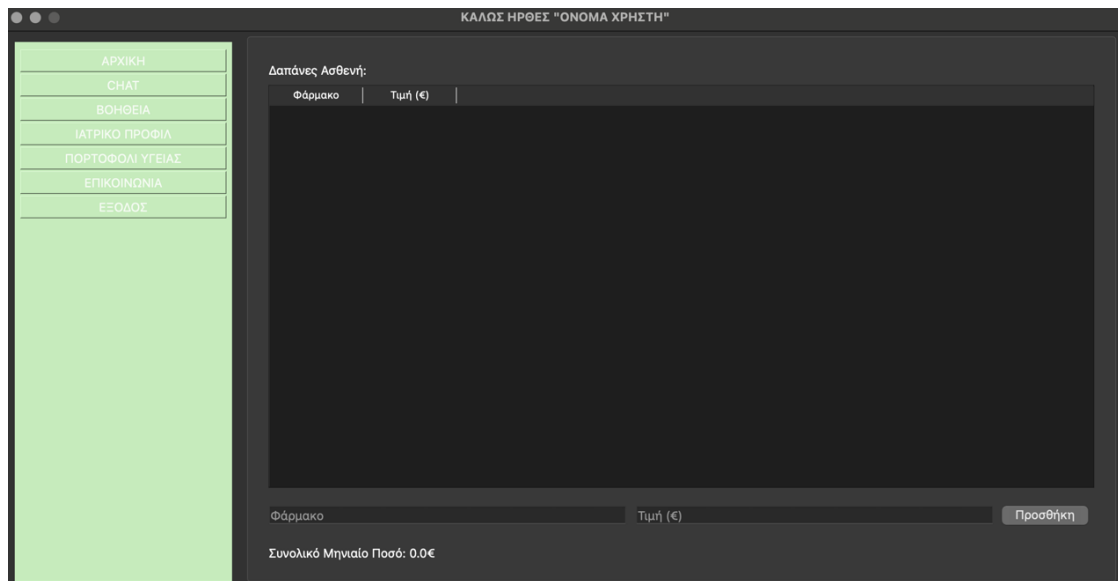
Πίεση:

Ομάδα Αίματος:

Χρώμα Ματιών:

ΕΠΕΞΕΡΓΑΣΙΑ ΔΕΔΟΜΕΝΩΝ

Με το κουμπί ΠΟΡΤΟΦΟΛΙ ΥΓΕΙΑΣ ο πολίτης θα μπορεί να ελέγξει τα ιατρικά του έξοδα και να υπολογίζεται αυτόματα κάθε δαπάνη του μηνιαίος, εφόσον ο πολίτης προσθέτει κάθε φορά οτιδήποτε ξοδεύει. Έτσι, θα γνωρίζει ανά πάσα ώρα και στιγμή πόσα έχει ξοδέψει για τα φάρμακά του:



Κώδικας:

```
import sys
from PyQt5.QtWidgets import (QApplication, QMainWindow, QPushButton, QVBoxLayout,
                              QWidget,
                              QHBoxLayout, QLabel, QLineEdit, QTextEdit, QTabWidget, QMenuBar,
                              QMessageBox,
                              QTableWidgetItem, QTableWidgetItem)
from PyQt5.QtGui import QIcon
from PyQt5.QtCore import Qt

class ContentWidget(QWidget):
    def __init__(self, content):
        super().__init__()
        self.layout = QVBoxLayout()
        self.label = QLabel(content, self)
        self.label.setAlignment(Qt.AlignCenter)
        self.layout.addWidget(self.label)
        self.setLayout(self.layout)

class MedicalProfileWidget(QWidget):
    def __init__(self):
        super().__init__()
        self.layout = QVBoxLayout()

        # Create labels and line edits for user details
        self.first_name_label = QLabel("Όνομα:", self)
        self.first_name_edit = QLineEdit("Γιάννης", self)

        self.last_name_label = QLabel("Επώνυμο:", self)
        self.last_name_edit = QLineEdit("Παπαδόπουλος", self)

        self.age_label = QLabel("Ηλικία:", self)
        self.age_edit = QLineEdit("30", self)

        self.weight_label = QLabel("Βάρος:", self)
```

```

self.weight_edit = QLineEdit("70kg", self)

self.height_label = QLabel("Ύψος:", self)
self.height_edit = QLineEdit("180cm", self)

self.pressure_label = QLabel("Πίεση:", self)
self.pressure_edit = QLineEdit("120/80", self)

self.blood_group_label = QLabel("Ομάδα Αίματος:", self)
self.blood_group_edit = QLineEdit("O+", self)

self.eye_color_label = QLabel("Χρώμα Ματιών:", self)
self.eye_color_edit = QLineEdit("Καστανά", self)

# Add widgets to the layout
self.layout.addWidget(self.first_name_label)
self.layout.addWidget(self.first_name_edit)
self.layout.addWidget(self.last_name_label)
self.layout.addWidget(self.last_name_edit)
self.layout.addWidget(self.age_label)
self.layout.addWidget(self.age_edit)
self.layout.addWidget(self.weight_label)
self.layout.addWidget(self.weight_edit)
self.layout.addWidget(self.height_label)
self.layout.addWidget(self.height_edit)
self.layout.addWidget(self.pressure_label)
self.layout.addWidget(self.pressure_edit)
self.layout.addWidget(self.blood_group_label)
self.layout.addWidget(self.blood_group_edit)
self.layout.addWidget(self.eye_color_label)
self.layout.addWidget(self.eye_color_edit)

# Add edit button
self.edit_button = QPushButton("ΕΠΕΞΕΡΓΑΣΙΑ ΔΕΔΟΜΕΝΩΝ", self)
self.edit_button.clicked.connect(self.save_data)
self.layout.addWidget(self.edit_button)

self.setLayout(self.layout)

def save_data(self):
    # Get data from line edits and do something with it (e.g., save to a file or database)
    first_name = self.first_name_edit.text()
    last_name = self.last_name_edit.text()
    age = self.age_edit.text()
    weight = self.weight_edit.text()
    height = self.height_edit.text()
    pressure = self.pressure_edit.text()
    blood_group = self.blood_group_edit.text()
    eye_color = self.eye_color_edit.text()

    # Here you can perform actions with the retrieved data, such as saving it to a file or
    database
    with open('medical_profile.txt', 'w') as file:
        file.write(f"First Name: {first_name}\n")
        file.write(f>Last Name: {last_name}\n")
        file.write(f"Age: {age}\n")
        file.write(f"Weight: {weight}\n")
        file.write(f"Height: {height}\n")
        file.write(f"Pressure: {pressure}\n")
        file.write(f"Blood Group: {blood_group}\n")

```

```

        file.write(f"Eye Color: {eye_color}\n")

# Optionally, you can display a message to confirm that the data has been saved
QMessageBox.information(self, "Αποθήκευση Δεδομένων", "Τα δεδομένα
αποθηκεύτηκαν επιτυχώς.")

# You can also clear the line edits after saving the data if needed
self.clear_line_edits()

def clear_line_edits(self):
    # Clear all line edits
    self.first_name_edit.clear()
    self.last_name_edit.clear()
    self.age_edit.clear()
    self.weight_edit.clear()
    self.height_edit.clear()
    self.pressure_edit.clear()
    self.blood_group_edit.clear()
    self.eye_color_edit.clear()

class HealthWalletWidget(QWidget):
    def __init__(self):
        super().__init__()
        self.layout = QVBoxLayout()

# Create a label and table for displaying expenses
self.expenses_label = QLabel("Δαπάνες Ασθενή:", self)
self.expenses_table = QTableWidgetItem(0, 2, self)
self.expenses_table.setHorizontalHeaderLabels(["Φάρμακο", "Τιμή (€)"])

# Create inputs for new expense
self.new_expense_layout = QHBoxLayout()
self.med_name_input = QLineEdit(self)
self.med_name_input.setPlaceholderText("Φάρμακο")
self.med_price_input = QLineEdit(self)
self.med_price_input.setPlaceholderText("Τιμή (€)")
self.add_expense_button = QPushButton("Προσθήκη", self)
self.add_expense_button.clicked.connect(self.add_expense)
self.new_expense_layout.addWidget(self.med_name_input)
self.new_expense_layout.addWidget(self.med_price_input)
self.new_expense_layout.addWidget(self.add_expense_button)

# Create a label for total monthly expenses
self.total_label = QLabel("Συνολικό Μηνιαίο Ποσό: 0.0€", self)

# Add widgets to the layout
self.layout.addWidget(self.expenses_label)
self.layout.addWidget(self.expenses_table)
self.layout.addLayout(self.new_expense_layout)
self.layout.addWidget(self.total_label)

self.setLayout(self.layout)

def add_expense(self):
    med_name = self.med_name_input.text()
    med_price = self.med_price_input.text()

    if med_name and med_price:
        try:
            price = float(med_price)

```



```

        row_position = self.expenses_table.rowCount()
        self.expenses_table.insertRow(row_position)
        self.expenses_table.setItem(row_position, 0, QTableWidgetItem(med_name))
        self.expenses_table.setItem(row_position, 1, QTableWidgetItem(f"{{price:.2f}}"))

        self.update_total()

        self.med_name_input.clear()
        self.med_price_input.clear()
    except ValueError:
        QMessageBox.warning(self, "Σφάλμα", "Η τιμή πρέπει να είναι αριθμός.")
    else:
        QMessageBox.warning(self, "Σφάλμα", "Συμπληρώστε όλα τα πεδία.")

def update_total(self):
    total = 0.0
    for row in range(self.expenses_table.rowCount()):
        price_item = self.expenses_table.item(row, 1)
        if price_item:
            total += float(price_item.text())

    self.total_label.setText(f"Συνολικό Μηνιαίο Ποσό: {{total:.2f}}€")

class ChatWidget(QWidget):
    def __init__(self):
        super().__init__()
        self.layout = QVBoxLayout()

        self.chat_area = QTextEdit(self)
        self.chat_area.setReadOnly(True)

        self.input_layout = QHBoxLayout()
        self.message_input = QLineEdit(self)
        self.send_button = QPushButton("Αποστολή", self)
        self.send_button.clicked.connect(self.send_message)

        self.input_layout.addWidget(self.message_input)
        self.input_layout.addWidget(self.send_button)

        self.layout.addWidget(self.chat_area)
        self.layout.addLayout(self.input_layout)

        self.setLayout(self.layout)

    def send_message(self):
        message = self.message_input.text()
        if message:
            self.chat_area.append(f"Εσύ: {{message}}")
            self.message_input.clear()
            # Here you could also add the code to send the message to the server or handle it
            accordingly

class HelpWidget(QWidget):
    def __init__(self):
        super().__init__()
        self.layout = QVBoxLayout()

        self.help_label = QLabel("Βοήθεια", self)
        self.help_label.setAlignment(Qt.AlignCenter)
        self.help_label.setStyleSheet("font-size: 20px; font-weight: bold;")

```

```

        self.help_text = QTextEdit(self)
        self.help_text.setReadOnly(True)
        self.help_text.setText("Εδώ μπορείτε να βρείτε βοήθεια σχετικά με τη χρήση της εφαρμογής...\n\n"
                                "1. Για να περιηγηθείτε στην εφαρμογή, χρησιμοποιήστε το μενού στα αριστερά.\n"
                                "2. Στην καρτέλα ΙΑΤΡΙΚΟ ΠΡΟΦΙΛ, μπορείτε να δείτε και να επεξεργαστείτε τα ιατρικά σας στοιχεία.\n"
                                "3. Στην καρτέλα ΠΟΡΤΟΦΟΛΙ ΥΓΕΙΑΣ, μπορείτε να προσθέσετε και να δείτε τις δαπάνες σας.\n"
                                "4. Στην καρτέλα CHAT, μπορείτε να συνομιλήσετε με έναν σύμβουλο ή γιατρό.\n"
                                "5. Εάν χρειάζεστε επιπλέον βοήθεια, επικοινωνήστε μαζί μας μέσω της καρτέλας ΕΠΙΚΟΙΝΩΝΙΑ.")

        self.layout.addWidget(self.help_label)
        self.layout.addWidget(self.help_text)

    self.setLayout(self.layout)

class MainWindow(QMainWindow):
    def __init__(self):
        super().__init__()

        self.setWindowTitle('ΚΑΛΩΣ ΗΡΘΕΣ "ΟΝΟΜΑ ΧΡΗΣΤΗ"')

        # Set the application icon
        self.setWindowIcon(QIcon('app_icon.png')) # Make sure app_icon.png is in the same directory

        # Set the size of the window
        self.setFixedSize(1200, 600)

        # Create the main layout
        self.main_layout = QHBoxLayout()

        # Add the menu widget to the main layout
        self.menu_widget = self.create_menu()
        self.main_layout.addWidget(self.menu_widget)

        # Create right side with tabs
        self.right_widget = QTabWidget()
        self.right_widget.tabBar().setObjectName("mainTab")

        self.tab1 = self.create_home_page()
        self.tab2 = ChatWidget() # Use ChatWidget for the chat page
        self.tab3 = HelpWidget() # Use HelpWidget for the help page
        self.tab4 = MedicalProfileWidget() # Use MedicalProfileWidget for the medical profile page
        self.tab5 = HealthWalletWidget() # Use HealthWalletWidget for the health wallet page
        self.tab6 = ContentWidget("This is the contact page content.")

        self.right_widget.addTab(self.tab1, "")
        self.right_widget.addTab(self.tab2, "")
        self.right_widget.addTab(self.tab3, "")
        self.right_widget.addTab(self.tab4, "")
        self.right_widget.addTab(self.tab5, "")
        self.right_widget.addTab(self.tab6, "")

```

```

self.right_widget.setCurrentIndex(0)
self.right_widget.setStyleSheet("QTabBar::tab{width: 0; height: 0; margin: 0; padding: 0;
border: none;}")

self.main_layout.addWidget(self.right_widget)
self.main_layout.setStretch(0, 1)
self.main_layout.setStretch(1, 4)

main_widget = QWidget()
main_widget.setLayout(self.main_layout)
self.setCentralWidget(main_widget)

# Add the menu bar with About action
self.create_menubar()

def create_menu(self):
    menu_layout = QVBoxLayout()
    self.add_menu_button(menu_layout, 'ΑΡΧΙΚΗ', 0)
    self.add_menu_button(menu_layout, 'CHAT', 1)
    self.add_menu_button(menu_layout, 'ΒΟΗΘΕΙΑ', 2)
    self.add_menu_button(menu_layout, 'ΙΑΤΡΙΚΟ ΠΡΟΦΙΛ', 3)
    self.add_menu_button(menu_layout, 'ΠΟΡΤΟΦΟΛΙ ΥΓΕΙΑΣ', 4)
    self.add_menu_button(menu_layout, 'ΕΠΙΚΟΙΝΩΝΙΑ', 5)

    exit_button = QPushButton('ΕΞΟΔΟΣ')
    exit_button.clicked.connect(self.close_application)
    menu_layout.addWidget(exit_button)
    menu_layout.addStretch()

    menu_widget = QWidget()
    menu_widget.setLayout(menu_layout)
    menu_widget.setStyleSheet("background-color: #CDEAC0;")
    return menu_widget

def add_menu_button(self, menu_layout, text, index):
    button = QPushButton(text, self)
    button.clicked.connect(lambda: self.right_widget.setCurrentIndex(index))
    menu_layout.addWidget(button)

def create_home_page(self):
    home_widget = QWidget()
    home_layout = QVBoxLayout()

    greeting_label = QLabel("Welcome, ΟΝΟΜΑ ΧΡΗΣΤΗ!", self)
    greeting_label.setAlignment(Qt.AlignCenter)
    greeting_label.setStyleSheet("font-size: 20px; font-weight: bold;")
    home_layout.addWidget(greeting_label)

    bottom_layout = QHBoxLayout()

    left_box = QVBoxLayout()
    left_label = QLabel('Προσωπικά Στοιχεία', self)
    left_label.setAlignment(Qt.AlignCenter)
    left_label.setStyleSheet("font-size: 16px; font-weight: bold;")
    left_box.addWidget(left_label)

    self.name_label = QLabel('Όνοματεπώνυμο:', self)
    self.name_edit = QLineEdit(self)
    self.name_edit.setReadOnly(True)
    left_box.addWidget(self.name_label)

```

```

left_box.addWidget(self.name_edit)

self.email_label = QLabel('email:', self)
self.email_edit = QLineEdit(self)
self.email_edit.setReadOnly(True)
left_box.addWidget(self.email_label)
left_box.addWidget(self.email_edit)

self.phone_label = QLabel('Τηλέφωνο:', self)
self.phone_edit = QLineEdit(self)
self.phone_edit.setReadOnly(True)
left_box.addWidget(self.phone_label)
left_box.addWidget(self.phone_edit)

self.edit_button = QPushButton('Επεξεργασία', self)
self.edit_button.clicked.connect(self.enable_editing)
left_box.addWidget(self.edit_button)

self.save_button = QPushButton('Αποθήκευση', self)
self.save_button.clicked.connect(self.save_data)
self.save_button.setVisible(False) # initially hidden
left_box.addWidget(self.save_button)

left_box.addWidget(QTextEdit(self))

right_box = QVBoxLayout()
right_label = QLabel('Ειδοποιήσεις', self)
right_label.setAlignment(Qt.AlignCenter)
right_label.setStyleSheet("font-size: 16px; font-weight: bold;")
right_box.addWidget(right_label)
right_box.addWidget(QLineEdit(self))
right_box.addWidget(QTextEdit(self))

bottom_layout.addLayout(left_box)
bottom_layout.addLayout(right_box)

home_layout.addLayout(bottom_layout)

home_widget.setLayout(home_layout)
return home_widget

def enable_editing(self):
    self.name_edit.setReadOnly(False)
    self.email_edit.setReadOnly(False)
    self.phone_edit.setReadOnly(False)
    self.save_button.setVisible(True) # show the save button

def save_data(self):
    # Add code here to save the data
    self.name_edit.setReadOnly(True)
    self.email_edit.setReadOnly(True)
    self.phone_edit.setReadOnly(True)
    self.save_button.setVisible(False) # hide the save button again

def create_menubar(self):
    menubar = QMenuBar(self)
    self.setMenuBar(menubar)

def close_application(self):
    QApplication.instance().quit()

```

```
if __name__ == '__main__':  
    app = QApplication(sys.argv)  
    window = MainWindow()  
    window.show()  
    sys.exit(app.exec_())
```

Το κουμπί βοήθεια:

The screenshot shows the main menu of the 'Health Application'. On the left, there is a vertical list of buttons: ΑΡΧΙΚΗ, CHAT, ΒΟΗΘΕΙΑ, ΙΑΤΡΙΚΟ ΠΡΟΦΙΛ, ΠΟΡΤΟΦΟΛΙ ΥΓΕΙΑΣ, ΕΠΙΚΟΙΝΩΝΙΑ, and ΕΞΟΔΟΣ. The 'ΒΟΗΘΕΙΑ' button is highlighted. On the right, there are three input fields with the following labels: ΧΡΕΙΑΖΟΜΑΙ ΦΑΡΜΑΚΑ, ΨΑΧΝΩ ΓΙΑΤΡΟ, and ΘΕΛΩ ΝΑ ΔΩΣΩ ΑΙΜΑ.

Επιλέγοντας από το μενού το κουμπί ΒΟΗΘΕΙΑ εμφανίζονται στον πολίτη 3 επιλογές. Μπορεί είτε να αναζητήσει φάρμακο, είτε να αναζητήσει γιατρό είτε τέλος να κλείσει ραντεβού για αιμοδοσία.

- Έστω ότι επιλέξει ΧΡΕΙΑΖΟΜΑΙ ΦΑΡΜΑΚΟ

The screenshot shows the search screen for medicines. On the left, the same vertical list of buttons is present, with 'ΒΟΗΘΕΙΑ' highlighted. The main area contains a form with the following elements: a 'ΣΥΜΠΤΩΜΑΤΑ:' label followed by a text input field and a 'Πυρετός' dropdown menu; a 'ΑΠΑΙΤΕΙΤΑΙ ΣΥΝΤΑΓΗ;' label followed by two radio buttons labeled 'ΝΑΙ' and 'ΟΧΙ'; and a 'Search' text input field.

- Έστω ότι επιλέξει ότι χρειάζεται γιατρό

ΑΡΧΙΚΗ	ΤΙ ΨΑΧΝΕΤΕ;
CHAT	
ΒΟΗΘΕΙΑ	
ΙΑΤΡΙΚΟ ΠΡΟΦΙΛ	
ΠΟΡΤΟΦΟΛΙ ΥΓΕΙΑΣ	
ΕΠΙΚΟΙΝΩΝΙΑ	
ΕΞΟΔΟΣ	
	ΣΕ ΤΙ ΓΙΑΤΡΟ ΝΑ ΑΠΕΥΘΥΝΩ
	Ειδικότητα:
	Γενικός Ιατρός
	Περιοχή:
	Αθήνα
	Κατ' οίκον επίσκεψη:
	<input type="radio"/> ΝΑΙ <input type="radio"/> ΟΧΙ
	Search

Κώδικας:

```
import sys
from PyQt5.QtWidgets import QApplication, QMainWindow, QPushButton, QVBoxLayout,
QCalendarWidget, QWidget, QHBoxLayout, QLabel, QLineEdit, QTextEdit, QMessageBox,
QComboBox, QCheckBox, QRadioButton, QButtonGroup, QScrollArea, QFrame,
QGraphicsOpacityEffect
from PyQt5.QtCore import Qt
from PyQt5.QtCore import pyqtSignal
from PyQt5.QtCore import QDate
from PyQt5.QtWidgets import QPushButton

class ContentWidget(QWidget):
    def __init__(self, content):
        super().__init__()
        self.layout = QVBoxLayout()
        self.label = QLabel(content, self)
        self.label.setAlignment(Qt.AlignCenter)
        self.layout.addWidget(self.label)
        self.setLayout(self.layout)

class HelpWidget(QWidget):
    def __init__(self, parent):
        super().__init__(parent)
        self.layout = QVBoxLayout()

        self.button1 = QPushButton("ΧΡΕΙΑΖΟΜΑΙ ΦΑΡΜΑΚΑ")
        self.button2 = QPushButton("ΨΑΧΝΩ ΓΙΑΤΡΟ")
        self.button3 = QPushButton("ΘΕΛΩ ΝΑ ΔΩΣΩ ΑΙΜΑ")

        self.layout.addWidget(self.button1)
        self.layout.addWidget(self.button2)
        self.layout.addWidget(self.button3)

        self.button1.clicked.connect(parent.show_medicine_options)
        self.button2.clicked.connect(parent.show_doctor_search)
        self.button3.clicked.connect(lambda: parent.show_content("Θέλω Να Δώσω Αίμα",
```

```

"Περιεχόμενο για Θέλω Να Δώσω Αίμα"))

self.setLayout(self.layout)

class MedicineWidget(QWidget):
    # Define signal for when a medicine is selected
    medicine_selected = pyqtSignal(list)

    def __init__(self, parent):
        super().__init__(parent)
        self.layout = QVBoxLayout()

        # Συμπτώματα
        self.symptom_layout = QHBoxLayout()
        self.symptom_label = QLabel("ΣΥΜΠΤΩΜΑΤΑ:", self)
        self.symptom_input = QLineEdit(self)
        self.symptom_list = QComboBox(self)
        self.symptom_list.addItems(["Πυρετός", "Βήχας", "Πονοκέφαλος"]) # Προσθήκη λίστας
        συμπτωμάτων
        self.symptom_layout.addWidget(self.symptom_label)
        self.symptom_layout.addWidget(self.symptom_input)
        self.symptom_layout.addWidget(self.symptom_list)

        # Απαιτείται Συνταγή
        self.prescription_layout = QHBoxLayout()
        self.prescription_label = QLabel("ΑΠΑΙΤΕΙΤΑΙ ΣΥΝΤΑΓΗ:", self)
        self.prescription_yes = QRadioButton("ΝΑΙ", self)
        self.prescription_no = QRadioButton("ΟΧΙ", self)
        self.prescription_group = QButtonGroup(self)
        self.prescription_group.addButton(self.prescription_yes)
        self.prescription_group.addButton(self.prescription_no)
        self.prescription_layout.addWidget(self.prescription_label)
        self.prescription_layout.addWidget(self.prescription_yes)
        self.prescription_layout.addWidget(self.prescription_no)

        # Search Button
        self.search_button = QPushButton("Search", self)
        self.search_button.clicked.connect(self.perform_search)

        self.layout.addLayout(self.symptom_layout)
        self.layout.addLayout(self.prescription_layout)
        self.layout.addWidget(self.search_button)
        self.layout.addStretch()

        self.setLayout(self.layout)

    def perform_search(self):
        # Πάρτε τα συμπτώματα που έχει εισάγει ο χρήστης
        selected_symptom = self.symptom_list.currentText()
        user_input_symptom = self.symptom_input.text()

        # Προετοιμάστε τα συμπτώματα για αναζήτηση (συμπτώματα + τυχόν επιπρόσθετα
        συμπτώματα που έχει εισάγει ο χρήστης)
        symptoms = [selected_symptom]
        if user_input_symptom:
            symptoms.append(user_input_symptom)

        # Πραγματοποιήστε την πραγματική αναζήτηση φαρμάκων εδώ, χρησιμοποιώντας τα
        συμπτώματα

```

```

        # Για τώρα, θα απλώς εκπέμψουμε ένα προκαθορισμένο λίστα φαρμάκων ως
        παράδειγμα
        medicines = [
            {"name": "Παρακεταμόλη", "description": "Για την ελάφρυνση του πυρετού και της
            πονοκέφαλου",
            "prescription_required": False},
            {"name": "Αμοξικιλίνη", "description": "Αντιβιοτικό για τη θεραπεία βακτηριακών
            λοιμώξεων",
            "prescription_required": True},
            {"name": "Βουδεσονίδη", "description": "Για την αντιμετώπιση του βήχα",
            "prescription_required": False},
        ]

        # Εκπομπή των αποτελεσμάτων χρησιμοποιώντας το σήμα medicine_selected
        self.medicine_selected.emit(medicines)

class MedicineResultsWidget(QWidget):
    def __init__(self, medicines, parent=None):
        super().__init__(parent)
        self.layout = QVBoxLayout()

        for medicine in medicines:
            # Δημιουργία πλαισίου για κάθε φάρμακο
            medicine_frame = QFrame(self)
            medicine_frame.setFrameShape(QFrame.Box)
            medicine_frame.setLineWidth(2)
            medicine_layout = QVBoxLayout(medicine_frame)

            # Ετικέτα για το όνομα του φαρμάκου
            medicine_name_label = QLabel(f"{medicine['name']}", self)
            medicine_layout.addWidget(medicine_name_label)

            # Ετικέτα για την περιγραφή του φαρμάκου
            medicine_description_label = QLabel(f"{medicine['description']}", self)
            medicine_layout.addWidget(medicine_description_label)

            # Ετικέτα για την απαίτηση συνταγής
            prescription_required_label = QLabel(f"Απαιτεί συνταγή: {'Ναι' if
            medicine['prescription_required'] else 'Όχι'}", self)
            medicine_layout.addWidget(prescription_required_label)

            # Κουμπί για εμφάνιση περαιτέρω πληροφοριών
            select_button = QPushButton("Επιλογή", self)
            select_button.clicked.connect(lambda checked, med=medicine:
            self.show_medicine_details(med))
            medicine_layout.addWidget(select_button)

            # Προσθήκη του πλαισίου στην κύρια διάταξη
            self.layout.addWidget(medicine_frame)

        self.setLayout(self.layout)

    def show_medicine_details(self, medicine):
        # Εδώ μπορείτε να υλοποιήσετε τη λειτουργία που θέλετε όταν γίνει κλικ στο φάρμακο
        print("Εμφάνιση περαιτέρω πληροφοριών για το φάρμακο:", medicine['name'])

class ReviewWidget(QWidget):

```



```

def __init__(self, doctor, parent):
    super().__init__(parent)
    self.doctor = doctor
    self.layout = QVBoxLayout()

    # Τίτλος
    self.title_label = QLabel(f"Υποβολή Αξιολόγησης για {doctor['name']}", self)
    self.layout.addWidget(self.title_label)

    # Πεδίο Αξιολόγησης
    self.review_text = QTextEdit(self)
    self.review_text.setPlaceholderText("Γράψτε την αξιολόγησή σας εδώ...")
    self.layout.addWidget(self.review_text)

    # Επιλογή Αστεριών
    self.rating_label = QLabel("Βαθμολογία:", self)
    self.layout.addWidget(self.rating_label)
    self.star_layout = QHBoxLayout()
    self.star_buttons = []
    for i in range(5):
        star_button = QRadioButton(f"{i + 1} Αστέρια", self)
        self.star_buttons.append(star_button)
        self.star_layout.addWidget(star_button)
    self.layout.addLayout(self.star_layout)

    # Κουμπί Υποβολής
    self.submit_button = QPushButton("Υποβολή", self)
    self.submit_button.clicked.connect(self.submit_review)
    self.layout.addWidget(self.submit_button)

    self.setLayout(self.layout)

    def submit_review(self):
        review_text = self.review_text.toPlainText()
        rating = next((i + 1 for i, btn in enumerate(self.star_buttons) if btn.isChecked()), None)
        if review_text and rating:
            # Προσθήκη αξιολόγησης σε μια λίστα αξιολογήσεων (θα μπορούσε να είναι βάση
            # δεδομένων σε πραγματική εφαρμογή)
            self.parent().add_review(self.doctor, review_text, rating)
            self.parent().show_message("Ευχαριστούμε για την αξιολόγησή σας!")
        else:
            self.parent().show_message("Παρακαλώ συμπληρώστε την αξιολόγηση και επιλέξτε
            βαθμολογία.")

class ChatWidget(QWidget):
    def __init__(self, doctor, parent):
        super().__init__(parent)
        self.layout = QVBoxLayout()

        self.chat_label = QLabel(f"Chat με τον {doctor['name']}", self)
        self.layout.addWidget(self.chat_label)

        self.chat_history = QTextEdit(self)
        self.chat_history.setReadOnly(True)
        self.layout.addWidget(self.chat_history)

        self.chat_input = QLineEdit(self)
        self.chat_input.setPlaceholderText("Πληκτρολογήστε το μήνυμά σας εδώ...")
        self.layout.addWidget(self.chat_input)

        self.send_button = QPushButton("Αποστολή", self)

```

```

self.send_button.clicked.connect(self.send_message)
self.layout.addWidget(self.send_button)

self.setLayout(self.layout)

# Αυτόματη αποστολή πρώτου μηνύματος
self.chat_history.append(f"Dr. {doctor['name']}: Πώς μπορώ να σας βοηθήσω;")

def send_message(self):
    message = self.chat_input.text()
    if message:
        self.chat_history.append(f"Εσείς: {message}")
        self.chat_input.clear()

class AppointmentWidget(QWidget):
    def __init__(self, doctor, parent):
        super().__init__(parent)
        self.layout = QVBoxLayout()

        self.calendar_label = QLabel(f"Κλείσιμο Ραντεβού με τον {doctor['name']}", self)
        self.layout.addWidget(self.calendar_label)

        self.calendar = QCalendarWidget(self)
        self.calendar.setMinimumDate(QDate.currentDate())
        self.layout.addWidget(self.calendar)

        self.book_button = QPushButton("Κλείσιμο Ραντεβού", self)
        self.book_button.clicked.connect(self.book_appointment)
        self.layout.addWidget(self.book_button)

        self.setLayout(self.layout)

    def book_appointment(self):
        selected_date = self.calendar.selectedDate()
        QMessageBox.information(self, "Ραντεβού Κλεισμένο", f"Το ραντεβού σας με τον {self.parent().doctor['name']} έχει κλειστεί για τις {selected_date.toString()}")

class DoctorSearchWidget(QWidget):
    def __init__(self, parent):
        super().__init__(parent)
        self.parent = parent
        self.layout = QVBoxLayout()

        # Πλαίσιο: ΤΙ ΨΑΧΝΕΤΕ;
        self.search_label = QLabel("ΤΙ ΨΑΧΝΕΤΕ;", self)
        self.search_label.setStyleSheet(
            "color: green; padding: 5px; border: 2px solid black; border-radius: 5px; font-weight: bold;")
        self.search_label.setAlignment(Qt.AlignCenter)
        self.layout.addWidget(self.search_label)

        # Πλαίσιο: ΣΕ ΤΙ ΓΙΑΤΡΟ ΝΑ ΑΠΕΥΘΥΝΘΩ
        self.doctor_label = QLabel("ΣΕ ΤΙ ΓΙΑΤΡΟ ΝΑ ΑΠΕΥΘΥΝΘΩ", self)
        self.doctor_label.setStyleSheet("font-weight: bold; color: black;")
        self.doctor_label.setAlignment(Qt.AlignCenter)
        self.layout.addWidget(self.doctor_label)

        # Ειδικότητα
        self.specialty_label = QLabel("Ειδικότητα:", self)
        self.specialty_combo = QComboBox(self)

```

```

self.specialty_combo.addItem(["Γενικός Ιατρός", "Παιδίατρος", "Οδοντίατρος"])
self.layout.addWidget(self.specialty_label)
self.layout.addWidget(self.specialty_combo)

# Περιοχή
self.location_label = QLabel("Περιοχή:", self)
self.location_combo = QComboBox(self)
self.location_combo.addItem(["Αθήνα", "Θεσσαλονίκη", "Πάτρα"])
self.layout.addWidget(self.location_label)
self.layout.addWidget(self.location_combo)

# Κατ' οίκον επίσκεψη
self.home_visit_label = QLabel("Κατ' οίκον επίσκεψη:", self)
self.home_visit_yes = QRadioButton("ΝΑΙ", self)
self.home_visit_no = QRadioButton("ΟΧΙ", self)
self.home_visit_group = QButtonGroup(self)
self.home_visit_group.addButton(self.home_visit_yes)
self.home_visit_group.addButton(self.home_visit_no)
self.home_visit_layout = QHBoxLayout()
self.home_visit_layout.addWidget(self.home_visit_yes)
self.home_visit_layout.addWidget(self.home_visit_no)
self.layout.addWidget(self.home_visit_label)
self.layout.addLayout(self.home_visit_layout)

# Search Button
self.search_button = QPushButton("Search", self)
self.search_button.clicked.connect(self.perform_search)
self.layout.addWidget(self.search_button)

self.setLayout(self.layout)

def perform_search(self):
    specialty = self.specialty_combo.currentText()
    location = self.location_combo.currentText()
    home_visit = self.home_visit_yes.isChecked()
    # Dummy data for doctors
    doctors = [
        {"name": "Dr. John Doe", "specialty": "Γενικός Ιατρός", "rating": "★★★★", "location": "Αθήνα", "address": "Οδός Παράδειγμα 1", "availability": "Δευτέρα-Παρασκευή"},
        {"name": "Dr. Jane Smith", "specialty": "Παιδίατρος", "rating": "★★★☆☆", "location": "Θεσσαλονίκη", "address": "Οδός Παράδειγμα 2", "availability": "Τρίτη-Σάββατο"},
        {"name": "Dr. Alice Johnson", "specialty": "Οδοντίατρος", "rating": "★★★☆☆", "location": "Πάτρα", "address": "Οδός Παράδειγμα 3", "availability": "Δευτέρα-Τετάρτη"},
    ]
    self.parent.show_doctor_results(doctors)

class DoctorActionsWidget(QWidget):
    def __init__(self, doctor, parent=None):
        super().__init__(parent)
        self.layout = QHBoxLayout()

        self.phone_button = QPushButton("Τηλέφωνο", self)
        self.phone_button.clicked.connect(lambda: self.call_doctor(doctor)) # Connect phone
button

        self.chat_button = QPushButton("Τσάτ", self)
        self.chat_button.clicked.connect(lambda: self.start_chat_with_doctor(doctor)) # Connect
chat button

        self.appointment_button = QPushButton("Ραντεβού", self)

```

```

        self.appointment_button.clicked.connect(lambda:
self.book_appointment_with_doctor(doctor)) # Connect appointment button

        self.layout.addWidget(self.phone_button)
        self.layout.addWidget(self.chat_button)
        self.layout.addWidget(self.appointment_button)

        self.setLayout(self.layout)

def call_doctor(self, doctor):
    # Πραγματοποιήστε την κλήση του γιατρού χρησιμοποιώντας τον αριθμό τηλεφώνου
    phone_number = doctor.get('phone_number', None)
    if phone_number:
        # Εδώ θα μπορούσατε να υλοποιήσετε τη λειτουργία κλήσης
        pass

def start_chat_with_doctor(self, doctor):
    # Εδώ θα μπορούσατε να ξεκινήσετε ένα τσάτ με τον γιατρό
    pass

def book_appointment_with_doctor(self, doctor):
    # Εδώ θα μπορούσατε να εμφανίσετε ένα ημερολόγιο για τον χρήστη να επιλέξει μια
    ημερομηνία
    pass

class DoctorDetailsWidget(QWidget):
    def __init__(self, doctor, parent):
        super().__init__(parent)
        self.layout = QVBoxLayout()
        self.setWindowFlag(Qt.FramelessWindowHint) # Remove window frame
        self.setStyleSheet("background-color: rgba(255, 255, 255, 0.95); border-radius: 15px;
padding: 20px;")

        # Doctor details
        self.name_label = QLabel(f"Όνομα: {doctor['name']}", self)
        self.specialty_label = QLabel(f"Ειδικότητα: {doctor['specialty']}", self)
        self.location_label = QLabel(f"Περιοχή: {doctor['location']}", self)
        self.address_label = QLabel(f"Διεύθυνση Γραφείου: {doctor['address']}", self)
        self.availability_label = QLabel(f"Διαθεσιμότητα Παντεβού: {doctor['availability']}", self)
        self.rating_label = QLabel(f"Αξιολογήσεις: {doctor['rating']}", self)

        self.layout.addWidget(self.name_label)
        self.layout.addWidget(self.specialty_label)
        self.layout.addWidget(self.location_label)
        self.layout.addWidget(self.address_label)
        self.layout.addWidget(self.availability_label)
        self.layout.addWidget(self.rating_label)

        # Interaction buttons
        self.button_layout = QHBoxLayout()
        self.phone_button = QPushButton("Τηλέφωνο", self)
        self.phone_button.clicked.connect(lambda: self.call_doctor(doctor)) # Connect phone
button
        self.phone_button.setCheckable(True) # Κάνει το κουμπί επιλέξιμο
        self.chat_button = QPushButton("Τσάτ", self)
        self.chat_button.clicked.connect(lambda: self.start_chat_with_doctor(doctor)) # Connect
chat button
        self.chat_button.setCheckable(True) # Κάνει το κουμπί επιλέξιμο
        self.appointment_button = QPushButton("Παντεβού", self)
        self.appointment_button.clicked.connect(lambda:

```

```

self.book_appointment_with_doctor(doctor)) # Connect appointment button
self.appointment_button.setCheckable(True) # Κάνει το κουμπί επιλέξιμο

self.button_layout.addWidget(self.phone_button)
self.button_layout.addWidget(self.chat_button)
self.button_layout.addWidget(self.appointment_button)
self.layout.addLayout(self.button_layout)

self.setLayout(self.layout)

# Add DoctorActionsWidget
self.actions_widget = DoctorActionsWidget(doctor, self)
self.layout.addWidget(self.actions_widget)

self.setLayout(self.layout)
class DoctorResultsWidget(QWidget):
    def __init__(self, doctors, parent):
        super().__init__(parent)
        self.parent = parent
        self.layout = QVBoxLayout()

        self.scroll_area = QScrollArea(self)
        self.scroll_area.setWidgetResizable(True)
        self.scroll_content = QWidget(self.scroll_area)
        self.scroll_layout = QVBoxLayout(self.scroll_content)

        for doctor in doctors:
            doctor_frame = QFrame(self)
            doctor_frame setFrameShape(QFrame.Box)
            doctor_frame.setLineWidth(2)
            doctor_layout = QVBoxLayout(doctor_frame)

            doctor_name_label = QLabel(f"Όνομα: {doctor['name']}", self)
            doctor_specialty_label = QLabel(f"Ειδικότητα: {doctor['specialty']}", self)
            doctor_rating_label = QLabel(f"Αξιολογήσεις: {doctor['rating']}", self)

            select_button = QPushButton("Επιλογή", self)
            select_button.clicked.connect(lambda checked, d=doctor:
self.parent.show_doctor_details(d))

            doctor_layout.addWidget(doctor_name_label)
            doctor_layout.addWidget(doctor_specialty_label)
            doctor_layout.addWidget(doctor_rating_label)
            doctor_layout.addWidget(select_button)

            self.scroll_layout.addWidget(doctor_frame)

        self.scroll_content.setLayout(self.scroll_layout)
        self.scroll_area.setWidget(self.scroll_content)
        self.layout.addWidget(self.scroll_area)

        self.setLayout(self.layout)

class MainWindow(QMainWindow):
    def __init__(self):
        super().__init__()
        self.setWindowTitle("Health Application")
        self.setGeometry(100, 100, 1200, 600)
        self.content_area = None

```

```

self.main_layout = QHBoxLayout()
self.menu_widget = self.create_menu()
self.main_layout.addWidget(self.menu_widget)

self.content_area = QWidget()
self.content_layout = QVBoxLayout()

self.greeting_label = QLabel("Welcome, ONOMA XPHΣTH!", self)
self.greeting_label.setAlignment(Qt.AlignCenter)
self.greeting_label.setStyleSheet("font-size: 20px; font-weight: bold;")
self.content_layout.addWidget(self.greeting_label)

self.bottom_layout = QHBoxLayout()
self.box1 = QLineEdit(self)
self.box2 = QTextEdit(self)
self.box1.setMinimumSize(200, 50)
self.box2.setMinimumSize(200, 50)
self.bottom_layout.addWidget(self.box1)
self.bottom_layout.addWidget(self.box2)
self.content_layout.addLayout(self.bottom_layout)

self.content_area.setLayout(self.content_layout)
self.main_layout.addWidget(self.content_area)

main_widget = QWidget()
main_widget.setLayout(self.main_layout)
self.setCentralWidget(main_widget)

self.setFixedSize(1200, 600)

def create_menu(self):
    menu_layout = QVBoxLayout()
    self.add_menu_button(menu_layout, 'ΑΡΧΙΚΗ', 'Home Page', 'This is the home page content.')
    self.add_menu_button(menu_layout, 'CHAT', 'Chat Page', 'This is the chat page content.')
    self.add_menu_button(menu_layout, 'ΒΟΗΘΕΙΑ', 'Help Page', 'This is the help page content.')
    self.add_menu_button(menu_layout, 'ΙΑΤΡΙΚΟ ΠΡΟΦΙΛ', 'Medical Profile Page', 'This is the medical profile page content.')
    self.add_menu_button(menu_layout, 'ΠΟΡΤΟΦΟΛΙ ΥΓΕΙΑΣ', 'Health Wallet Page', 'This is the health wallet page content.')
    self.add_menu_button(menu_layout, 'ΕΠΙΚΟΙΝΩΝΙΑ', 'Contact Page', 'This is the contact page content.')

    exit_button = QPushButton('ΕΞΟΔΟΣ')
    exit_button.clicked.connect(self.close_application)
    menu_layout.addWidget(exit_button)
    menu_layout.addStretch() # Add stretch to push menu items to the top

    menu_widget = QWidget()
    menu_widget.setLayout(menu_layout)
    menu_widget.setStyleSheet("background-color: #CDEAC0;")
    menu_widget.setMinimumWidth(self.frameGeometry().width() // 5)
    return menu_widget

def add_menu_button(self, menu_layout, text, window_title=None, window_content=None):
    button = QPushButton(text, self)
    if text == 'ΑΡΧΙΚΗ':

```

```

        button.clicked.connect(self.show_home_page)
    elif text == 'ΒΟΗΘΕΙΑ':
        button.clicked.connect(self.show_help_page)
    else:
        button.clicked.connect(lambda: self.show_content(window_title, window_content))
    menu_layout.addWidget(button)

def show_home_page(self):
    self.clear_content()
    self.content_area = QWidget()
    self.content_layout = QVBoxLayout()

    self.greeting_label = QLabel("Welcome, ONOMA XPHΣTH!", self)
    self.greeting_label.setAlignment(Qt.AlignCenter)
    self.greeting_label.setStyleSheet("font-size: 20px; font-weight: bold;")
    self.content_layout.addWidget(self.greeting_label)

    self.bottom_layout = QHBoxLayout()
    self.box1 = QLineEdit(self)
    self.box2 = QTextEdit(self)
    self.box1.setMinimumSize(200, 50)
    self.box2.setMinimumSize(200, 50)
    self.bottom_layout.addWidget(self.box1)
    self.bottom_layout.addWidget(self.box2)
    self.content_layout.addLayout(self.bottom_layout)

    self.content_area.setLayout(self.content_layout)
    self.main_layout.addWidget(self.content_area)

def show_help_page(self):
    self.clear_content()
    self.content_area = HelpWidget(self)
    self.main_layout.addWidget(self.content_area)

def show_medicine_options(self):
    self.clear_content()
    self.content_area = MedicineWidget(self)
    self.main_layout.addWidget(self.content_area)
    self.connect_medicine_widget(self.content_area)

def show_medicine_results(self, medicines):
    self.clear_content()
    self.content_area = MedicineWidget(medicines, self)
    # Connect the signal to show medicine details
    self.content_area.medicine_selected.connect(self.show_medicine_details)
    self.main_layout.addWidget(self.content_area)

def connect_medicine_widget(self, widget):
    widget.medicine_selected.connect(self.show_medicine_results)

def show_medicine_details(self, medicine):
    self.clear_content()
    self.content_area = MedicineWidget(medicine, self)
    self.main_layout.addWidget(self.content_area)

def show_doctor_search(self):
    self.clear_content()
    self.content_area = DoctorSearchWidget(self)
    self.main_layout.addWidget(self.content_area)

```

```

def show_doctor_results(self, doctors):
    self.clear_content()
    self.content_area = DoctorResultsWidget(doctors, self)
    self.main_layout.addWidget(self.content_area)

def show_doctor_details(self, doctor):
    self.clear_content()
    self.content_area = DoctorDetailsWidget(doctor, self)
    self.main_layout.addWidget(self.content_area)

def show_chat_widget(self, doctor):
    self.clear_main_layout()
    self.chat_widget = ChatWidget(doctor, self)
    self.main_layout.addWidget(self.chat_widget)

def show_phone_number(self, doctor):
    QMessageBox.information(self, "Αριθμός Τηλεφώνου", f"Ο αριθμός τηλεφώνου του {doctor['name']} είναι {doctor['phone']}")

def show_appointment_widget(self, doctor):
    self.clear_main_layout()
    self.appointment_widget = AppointmentWidget(doctor, self)
    self.main_layout.addWidget(self.appointment_widget)

def show_content(self, title, content):
    self.clear_content()
    self.content_area = ContentWidget(content)
    self.main_layout.addWidget(self.content_area)

def clear_content(self):
    if self.content_area:
        self.main_layout.removeWidget(self.content_area)
        self.content_area.deleteLater()
        self.content_area = None

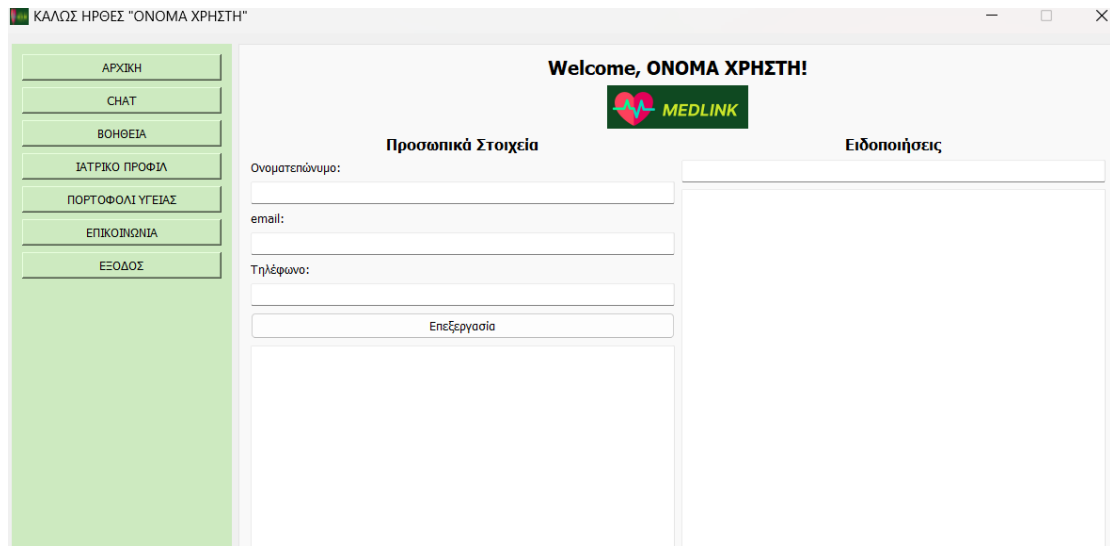
def close_application(self):
    QApplication.instance().quit()

def show_medicine_results(self, medicines):
    self.clear_content()
    self.content_area = MedicineResultsWidget(medicines, self)
    self.main_layout.addWidget(self.content_area)

if __name__ == "__main__":
    app = QApplication(sys.argv)
    main_window = MainWindow()
    main_window.show()
    sys.exit(app.exec_())

```


Ορισμός εικόνας εφαρμογής:



Κώδικας:

```
import sys
import os
from PyQt5.QtWidgets import QApplication, QMainWindow, QPushButton, QVBoxLayout,
QWidget, QHBoxLayout, QLabel, QLineEdit, QTextEdit, QTabWidget, QMenuBar
from PyQt5.QtGui import QIcon, QPixmap
from PyQt5.QtCore import Qt

class ContentWidget(QWidget):
    def __init__(self, content):
        super().__init__()
        self.layout = QVBoxLayout()
        self.label = QLabel(content, self)
        self.label.setAlignment(Qt.AlignCenter)
        self.layout.addWidget(self.label)
        self.setLayout(self.layout)

class MainWindow(QMainWindow):
    def __init__(self):
        super().__init__()

        self.setWindowTitle('ΚΑΛΩΣ ΗΡΘΕΣ "ΟΝΟΜΑ ΧΡΗΣΤΗ"')

        # Set the application icon
        icon_path = os.path.abspath('medlinkicon.jpg')
        if not os.path.exists(icon_path):
            print(f'Icon file not found: {icon_path}')
```

```

else:
    print(f"Icon file found: {icon_path}")
    self.setWindowIcon(QIcon(icon_path)) # Make sure medlinkicon.jpg is in the same
directory

# Set the size of the window
self.setFixedSize(1200, 600)

# Create the main layout
self.main_layout = QHBoxLayout()

# Add the menu widget to the main layout
self.menu_widget = self.create_menu()
self.main_layout.addWidget(self.menu_widget)

# Create right side with tabs
self.right_widget = QTabWidget()
self.right_widget.tabBar().setObjectName("mainTab")

self.tab1 = self.create_home_page()
self.tab2 = ContentWidget("This is the chat page content.")
self.tab3 = ContentWidget("This is the help page content.")
self.tab4 = ContentWidget("This is the medical profile page content.")
self.tab5 = ContentWidget("This is the health wallet page content.")
self.tab6 = ContentWidget("This is the contact page content.")

self.right_widget.addTab(self.tab1, "")
self.right_widget.addTab(self.tab2, "")
self.right_widget.addTab(self.tab3, "")
self.right_widget.addTab(self.tab4, "")
self.right_widget.addTab(self.tab5, "")
self.right_widget.addTab(self.tab6, "")

self.right_widget.setCurrentIndex(0)
self.right_widget.setStyleSheet("""QTabBar::tab{width: 0; height: 0; margin: 0; padding: 0;
border: none;}""")

self.main_layout.addWidget(self.right_widget)
self.main_layout.setStretch(0, 1)
self.main_layout.setStretch(1, 4)

main_widget = QWidget()
main_widget.setLayout(self.main_layout)
self.setCentralWidget(main_widget)

self.create_menubar()

#δημιουργία μενου του civilian
def create_menu(self):
    menu_layout = QVBoxLayout()
    self.add_menu_button(menu_layout, 'ΑΡΧΙΚΗ', 0)
    self.add_menu_button(menu_layout, 'CHAT', 1)
    self.add_menu_button(menu_layout, 'ΒΟΗΘΕΙΑ', 2)
    self.add_menu_button(menu_layout, 'ΙΑΤΡΙΚΟ ΠΡΟΦΙΛ', 3)
    self.add_menu_button(menu_layout, 'ΠΟΡΤΟΦΟΛΙ ΥΓΕΙΑΣ', 4)
    self.add_menu_button(menu_layout, 'ΕΠΙΚΟΙΝΩΝΙΑ', 5)

    exit_button = QPushButton('ΕΞΟΔΟΣ')
    exit_button.clicked.connect(self.close_application)
    menu_layout.addWidget(exit_button)

```

```

        menu_layout.addStretch()

        menu_widget = QWidget()
        menu_widget.setLayout(menu_layout)
        menu_widget.setStyleSheet("background-color: #CDEAC0;")
        return menu_widget

    def add_menu_button(self, menu_layout, text, index):
        button = QPushButton(text, self)
        button.clicked.connect(lambda: self.right_widget.setCurrentIndex(index))
        menu_layout.addWidget(button)

    def create_home_page(self):
        home_widget = QWidget()
        home_layout = QVBoxLayout()

        greeting_label = QLabel('Welcome, ONOMA XPHΣTH!', self)
        greeting_label.setAlignment(Qt.AlignCenter)
        greeting_label.setStyleSheet("font-size: 20px; font-weight: bold;")
        home_layout.addWidget(greeting_label)

        # Add image to home page
        image_label = QLabel(self)
        pixmap = QPixmap('medlinkicon.jpg') # Replace 'medlinkicon.jpg' with your image file
name
        if pixmap.isNull():
            print("Failed to load image")
        else:
            scaled_pixmap = pixmap.scaled(150, 150, Qt.KeepAspectRatio,
Qt.SmoothTransformation) # Adjust the size as needed
            image_label.setPixmap(scaled_pixmap)
            image_label.setAlignment(Qt.AlignCenter) # Center the image if desired
            home_layout.addWidget(image_label)

        bottom_layout = QHBoxLayout()

        left_box = QVBoxLayout()
        left_label = QLabel('Προσωπικά Στοιχεία', self)
        left_label.setAlignment(Qt.AlignCenter)
        left_label.setStyleSheet("font-size: 16px; font-weight: bold;")
        left_box.addWidget(left_label)

        self.name_label = QLabel('Όνοματεπώνυμο:', self)
        self.name_edit = QLineEdit(self)
        self.name_edit.setReadOnly(True)
        left_box.addWidget(self.name_label)
        left_box.addWidget(self.name_edit)

        self.email_label = QLabel('email:', self)
        self.email_edit = QLineEdit(self)
        self.email_edit.setReadOnly(True)
        left_box.addWidget(self.email_label)
        left_box.addWidget(self.email_edit)

        self.phone_label = QLabel('Τηλέφωνο:', self)
        self.phone_edit = QLineEdit(self)
        self.phone_edit.setReadOnly(True)
        left_box.addWidget(self.phone_label)
        left_box.addWidget(self.phone_edit)

```

```

self.edit_button = QPushButton('Επεξεργασία', self)
self.edit_button.clicked.connect(self.enable_editing)
left_box.addWidget(self.edit_button)

self.save_button = QPushButton('Αποθήκευση', self)
self.save_button.clicked.connect(self.save_data)
self.save_button.setVisible(False) # initially hidden
left_box.addWidget(self.save_button)

left_box.addWidget(QTextEdit(self))

right_box = QVBoxLayout()
right_label = QLabel('Ειδοποιήσεις', self)
right_label.setAlignment(Qt.AlignCenter)
right_label.setStyleSheet("font-size: 16px; font-weight: bold;")
right_box.addWidget(right_label)
right_box.addWidget(QLineEdit(self))
right_box.addWidget(QTextEdit(self))

bottom_layout.addLayout(left_box)
bottom_layout.addLayout(right_box)

home_layout.addLayout(bottom_layout)

home_widget.setLayout(home_layout)
return home_widget

def enable_editing(self):
    self.name_edit.setReadOnly(False)
    self.email_edit.setReadOnly(False)
    self.phone_edit.setReadOnly(False)
    self.save_button.setVisible(True) # show the save button

def save_data(self):
    # Add code here to save the data
    self.name_edit.setReadOnly(True)
    self.email_edit.setReadOnly(True)
    self.phone_edit.setReadOnly(True)
    self.save_button.setVisible(False) # hide the save button again

def create_menubar(self):
    menubar = QMenuBar(self)
    self.setMenuBar(menubar)

def close_application(self):
    QApplication.instance().quit()

if __name__ == '__main__':
    app = QApplication(sys.argv)
    window = MainWindow()
    window.show()
    sys.exit(app.exec_())

```

Περιβάλλον Γιατρού:

Η αρχική σελίδα του γιατρού μοιάζει αρκετά με αυτή του πολίτη απλά έχουμε διαφορετικές επιλογές ως μενού καθώς είναι διαφορετικό είδος χρήστη.

ΑΡΧΙΚΗ

CHAT

ΗΜΕΡΟΛΟΓΙΟ ΡΑΝΤΕΒΟΥ

ΣΥΝΤΑΓΟΓΡΑΦΗΣΗ

ΑΤΖΕΝΤΑ ΑΣΘΕΝΩΝ

ΕΠΙΚΟΙΝΩΝΙΑ

ΕΞΟΔΟΣ

Welcome, ΟΝΟΜΑ ΧΡΗΣΤΗ!

Προσωπικά Στοιχεία

Όνοματεπώνυμο:

email:

Τηλέφωνο:

Επεξεργασία

Ειδοποιήσεις

Για την ώρα έχουμε υλοποιήσει το κουμπί ΗΜΕΡΟΛΟΓΙΟ ΡΣΝΤΕΒΟΥ να εμφανίζει ένα ημερολόγιο

ΑΡΧΙΚΗ

CHAT

ΗΜΕΡΟΛΟΓΙΟ ΡΑΝΤΕΒΟΥ

ΣΥΝΤΑΓΟΓΡΑΦΗΣΗ

ΑΤΖΕΝΤΑ ΑΣΘΕΝΩΝ

ΕΠΙΚΟΙΝΩΝΙΑ

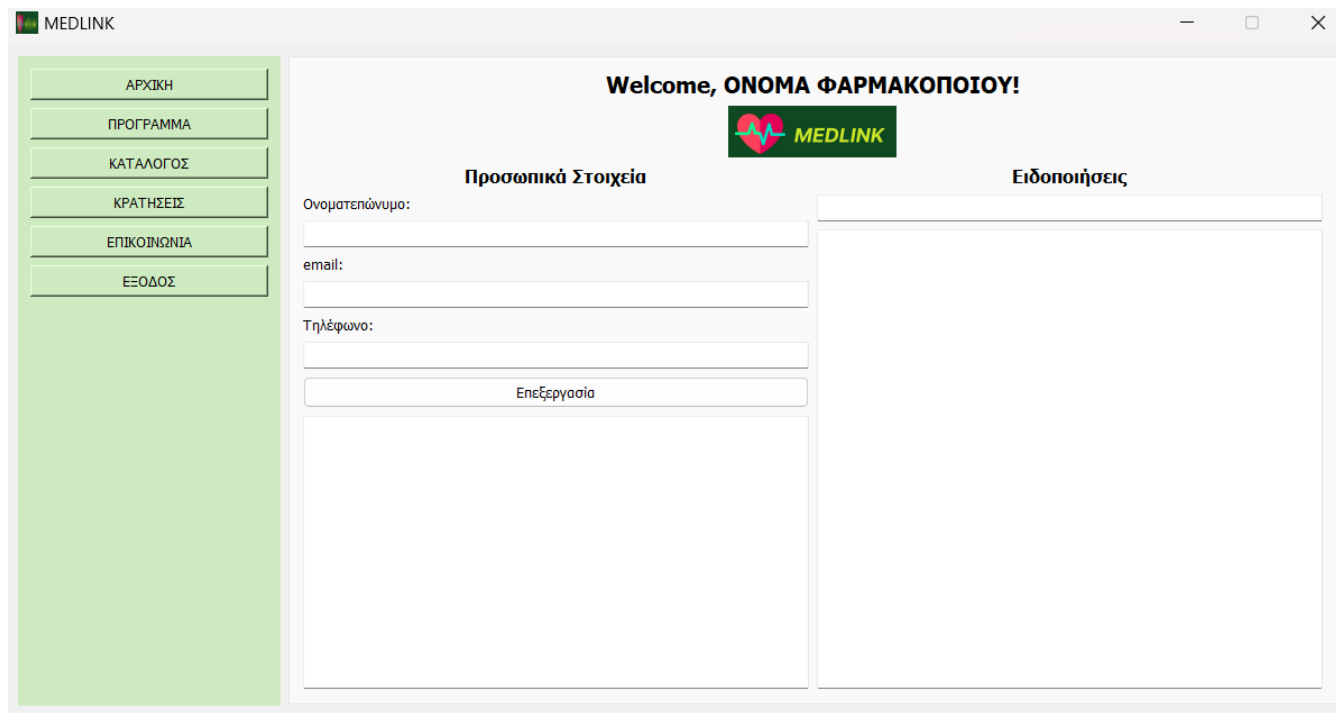
ΕΞΟΔΟΣ

Μάιος 2024

	Δευ	Τρι	Τετ	Περ	Παρ	Σαβ	Κυρ
18	29	30	1	2	3	4	5
19	6	7	8	9	10	11	12
20	13	14	15	16	17	18	19
21	20	21	22	23	24	25	26
22	27	28	29	30	31	1	2
23	3	4	5	6	7	8	9

Προσθήκες:

Περιβάλλον Φαρμακοποιού:



```
import sys
import os
from PyQt5.QtWidgets import QApplication, QMainWindow, QPushButton, QVBoxLayout,
QWidget, QHBoxLayout, QLabel, QLineEdit, QTextEdit, QTabWidget, QMenuBar,
QCalendarWidget
from PyQt5.QtGui import QIcon, QPixmap
from PyQt5.QtCore import Qt

class ContentWidget(QWidget):
    def __init__(self, content):
        super().__init__()
        self.layout = QVBoxLayout()
        self.label = QLabel(content, self)
        self.label.setAlignment(Qt.AlignCenter)
        self.layout.addWidget(self.label)
        self.setLayout(self.layout)

class MainWindow(QMainWindow):
    def __init__(self):
        super().__init__()

        self.setWindowTitle('MEDLINK')

        # Set the application icon
        icon_path = os.path.abspath('medlinkicon.jpg')
        if not os.path.exists(icon_path):
            print(f"Icon file not found: {icon_path}")
```

```

else:
    print(f"Icon file found: {icon_path}")
    self.setWindowIcon(QIcon(icon_path)) # Make sure medlinkicon.jpg is in the
same directory

# Set the size of the window
self.setFixedSize(1200, 600)

# Create the main layout
self.main_layout = QHBoxLayout()

# Add the menu widget to the main layout
self.menu_widget = self.create_menu()
self.main_layout.addWidget(self.menu_widget)

# Create right side with tabs
self.right_widget = QTabWidget()
self.right_widget.tabBar().setObjectName("mainTab")

self.tab1 = self.create_home_page()
self.tab2 = ContentWidget("This is the chat page content.")
self.tab3 = ContentWidget("Yep")
self.tab4 = ContentWidget("This is the prescription page content.")
self.tab5 = ContentWidget("This is the patient agenda page content.")
self.tab6 = ContentWidget("This is the contact page content.")

self.right_widget.addTab(self.tab1, '')
self.right_widget.addTab(self.tab2, '')
self.right_widget.addTab(self.tab3, '')
self.right_widget.addTab(self.tab4, '')
self.right_widget.addTab(self.tab5, '')
self.right_widget.addTab(self.tab6, '')

self.right_widget.setCurrentIndex(0)
self.right_widget.setStyleSheet('''QTabBar::tab{width: 0; height: 0;
margin: 0; padding: 0; border: none;}''')

self.main_layout.addWidget(self.right_widget)
self.main_layout.setStretch(0, 1)
self.main_layout.setStretch(1, 4)

main_widget = QWidget()
main_widget.setLayout(self.main_layout)
self.setCentralWidget(main_widget)

# Add the menu bar with About action
self.create_menubar()

def create_menu(self):
    menu_layout = QVBoxLayout()
    self.add_menu_button(menu_layout, 'ΑΡΧΙΚΗ', 0)
    self.add_menu_button(menu_layout, 'ΠΡΟΓΡΑΜΜΑ', 1)
    self.add_menu_button(menu_layout, 'ΚΑΤΑΛΟΓΟΣ', 2)
    self.add_menu_button(menu_layout, 'ΚΡΑΤΗΣΕΙΣ', 3)
    self.add_menu_button(menu_layout, 'ΕΠΙΚΟΙΝΩΝΙΑ', 4)

    exit_button = QPushButton('ΕΞΟΔΟΣ')
    exit_button.clicked.connect(self.close_application)
    menu_layout.addWidget(exit_button)
    menu_layout.addStretch()

    menu_widget = QWidget()
    menu_widget.setLayout(menu_layout)
    menu_widget.setStyleSheet("background-color: #CDEAC0;")

```

```

        return menu_widget

    def add_menu_button(self, menu_layout, text, index):
        button = QPushButton(text, self)
        button.clicked.connect(lambda: self.right_widget.setCurrentIndex(index))
        menu_layout.addWidget(button)

    def create_home_page(self):
        home_widget = QWidget()
        home_layout = QVBoxLayout()

        greeting_label = QLabel('Welcome, ΟΝΟΜΑ ΦΑΡΜΑΚΟΠΟΙΟΥ!', self)
        greeting_label.setAlignment(Qt.AlignCenter)
        greeting_label.setStyleSheet("font-size: 20px; font-weight: bold;")
        home_layout.addWidget(greeting_label)

        # Add image to home page
        image_label = QLabel(self)
        pixmap = QPixmap('medlinkicon.jpg') # Replace 'medlinkicon.jpg' with your
image file name
        if pixmap.isNull():
            print("Failed to load image")
        else:
            scaled_pixmap = pixmap.scaled(150, 150, Qt.KeepAspectRatio,
Qt.SmoothTransformation) # Adjust the size as needed
            image_label.setPixmap(scaled_pixmap)
            image_label.setAlignment(Qt.AlignCenter) # Center the image if desired
            home_layout.addWidget(image_label)

        bottom_layout = QHBoxLayout()

        left_box = QVBoxLayout()
        left_label = QLabel('Προσωπικά Στοιχεία', self)
        left_label.setAlignment(Qt.AlignCenter)
        left_label.setStyleSheet("font-size: 16px; font-weight: bold;")
        left_box.addWidget(left_label)

        self.name_label = QLabel('Όνοματεπώνυμο:', self)
        self.name_edit = QLineEdit(self)
        self.name_edit.setReadOnly(True)
        left_box.addWidget(self.name_label)
        left_box.addWidget(self.name_edit)

        self.email_label = QLabel('email:', self)
        self.email_edit = QLineEdit(self)
        self.email_edit.setReadOnly(True)
        left_box.addWidget(self.email_label)
        left_box.addWidget(self.email_edit)

        self.phone_label = QLabel('Τηλέφωνο:', self)
        self.phone_edit = QLineEdit(self)
        self.phone_edit.setReadOnly(True)
        left_box.addWidget(self.phone_label)
        left_box.addWidget(self.phone_edit)

        self.edit_button = QPushButton('Επεξεργασία', self)
        self.edit_button.clicked.connect(self.enable_editing)
        left_box.addWidget(self.edit_button)

        self.save_button = QPushButton('Αποθήκευση', self)
        self.save_button.clicked.connect(self.save_data)
        self.save_button.setVisible(False) # initially hidden
        left_box.addWidget(self.save_button)

```



```

        left_box.addWidget(QTextEdit(self))

        right_box = QVBoxLayout()
        right_label = QLabel('Ειδοποιήσεις', self)
        right_label.setAlignment(Qt.AlignCenter)
        right_label.setStyleSheet("font-size: 16px; font-weight: bold;")
        right_box.addWidget(right_label)
        right_box.addWidget(QLineEdit(self))
        right_box.addWidget(QTextEdit(self))

        bottom_layout.addLayout(left_box)
        bottom_layout.addLayout(right_box)

        home_layout.addLayout(bottom_layout)

        home_widget.setLayout(home_layout)
        return home_widget

    def enable_editing(self):
        self.name_edit.setReadOnly(False)
        self.email_edit.setReadOnly(False)
        self.phone_edit.setReadOnly(False)
        self.save_button.setVisible(True) # show the save button

    def save_data(self):
        # Add code here to save the data
        self.name_edit.setReadOnly(True)
        self.email_edit.setReadOnly(True)
        self.phone_edit.setReadOnly(True)
        self.save_button.setVisible(False) # hide the save button again

    def create_menubar(self):
        menubar = QMenuBar(self)
        self.setMenuBar(menubar)

    def close_application(self):
        QApplication.instance().quit()

if __name__ == '__main__':
    app = QApplication(sys.argv)
    window = MainWindow()
    window.show()
    sys.exit(app.exec_())


```

Για την εβδομάδα που είχαμε να ασχοληθούμε με το τέταρτο παραδοτέο, επειδή θεωρήσαμε ότι δεν χρειαζόταν να κάνουμε μεγάλες αλλαγές στα προηγούμενα προσπαθήσαμε να δημιουργήσουμε με κώδικα μια οθόνη ώστε να είναι αρκετά κοντά στα ενδεικτικά mock-up screens που έχουμε παρουσιάσει στο Project Description.

Λόγω περιορισμένου χρόνου επιλέξαμε να υλοποιήσουμε ένα σχετικά απλό Mock-Up Screen αλλά για τον ίδιο λόγο δεν καταφέραμε να το επεκτείνουμε και σε μεγαλύτερο μέρος της εφαρμογής μας. Το Mock-Up Screen που επιλέξαμε να υλοποιήσουμε είναι αυτό της κριτικής που βάζει ένας πολίτης σε ένα γιατρό.

Αυτό είναι το αρχικό:





Όνομα Γιατρού

Κάνε μια αξιολόγηση:

☆☆☆☆☆☆☆☆☆☆ /10

Σχόλια:

Υποβολή

Αυτό είναι με τον κώδικά μας:

Doctor Review

ΑΡΧΙΚΗ

CHAT

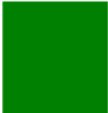
ΒΟΗΘΕΙΑ

ΙΑΤΡΙΚΟ ΠΡΟΦΙΛ

ΠΟΡΤΟΦΟΛΙ ΥΓΕΙΑΣ

ΕΠΙΚΟΙΝΩΝΙΑ

ΕΞΟΔΟΣ



Όνομα Γιατρού

Κάνε μια αξιολόγηση:

☆☆☆☆☆☆☆☆☆☆ /10

Σχόλια:

Υποβολή

Ο κώδικας είναι ο παρακάτω:

```
import tkinter as tk
from tkinter import ttk

# Function to handle submission
def submit_review():
    rating = rating_var.get()
    comments = comments_text.get("1.0", tk.END).strip()
    print(f"Submitted Rating: {rating}/10")
    print(f"Submitted Comments: {comments}")

# Create the main window
```

```

root = tk.Tk()
root.title("Doctor Review")
root.geometry("800x600")

# Create sidebar frame
sidebar = tk.Frame(root, bg='darkgreen', width=200)
sidebar.pack(side='left', fill='y')

# Add buttons to the sidebar
buttons = [
    ("ΑΡΧΙΚΗ", lambda: print("ΑΡΧΙΚΗ clicked")),
    ("CHAT", lambda: print("CHAT clicked")),
    ("ΒΟΗΘΕΙΑ", lambda: print("ΒΟΗΘΕΙΑ clicked")),
    ("ΙΑΤΡΙΚΟ ΠΡΟΦΙΛ", lambda: print("ΙΑΤΡΙΚΟ ΠΡΟΦΙΛ clicked")),
    ("ΠΟΡΤΟΦΟΛΙ ΥΓΕΙΑΣ", lambda: print("ΠΟΡΤΟΦΟΛΙ ΥΓΕΙΑΣ clicked")),
    ("ΕΠΙΚΟΙΝΩΝΙΑ", lambda: print("ΕΠΙΚΟΙΝΩΝΙΑ clicked")),
    ("ΕΞΟΔΟΣ", lambda: root.quit())
]

for text, command in buttons:
    btn = tk.Button(sidebar, text=text, bg='darkgreen', fg='white', command=command,
                    padx=10, pady=5, anchor='w')
    btn.pack(fill='x')

# Create main content frame
content = tk.Frame(root, bg='white')
content.pack(side='right', fill='both', expand=True)

# Doctor's information frame
doctor_frame = tk.Frame(content, bg='white', padx=20, pady=20)
doctor_frame.pack(pady=20, padx=20, fill='both', expand=True)

# Doctor's image placeholder
doctor_image = tk.Label(doctor_frame, text="", bg='green', width=10, height=5)
doctor_image.grid(row=0, column=0, rowspan=2, padx=20, pady=10)

# Doctor's name
doctor_name = tk.Label(doctor_frame, text="Όνομα Γιατρού", bg='white',
                        font=("Helvetica", 16))
doctor_name.grid(row=0, column=1, sticky='w', pady=10)

# Rating section
rating_label = tk.Label(doctor_frame, text="Κάνε μια αξιολόγηση:", bg='white',
                        font=("Helvetica", 12))
rating_label.grid(row=1, column=1, sticky='w', pady=10)

# Stars rating
rating_var = tk.IntVar()
stars_frame = tk.Frame(doctor_frame, bg='white')
stars_frame.grid(row=2, column=1, sticky='w', pady=10)

```

```

for i in range(10):
    star = tk.Radiobutton(stars_frame, text="★", variable=rating_var, value=i+1,
bg='white', fg='green', selectcolor='white', indicatoron=0)
    star.pack(side='left', padx=1)

rating_scale = tk.Label(stars_frame, text="/10", bg='white', font=("Helvetica", 12))
rating_scale.pack(side='left', padx=10)

# Comments section
comments_label = tk.Label(doctor_frame, text="Σχόλια:", bg='white', font=("Helvetica",
12))
comments_label.grid(row=3, column=0, sticky='nw', pady=10)

comments_text = tk.Text(doctor_frame, width=50, height=10, borderwidth=1,
relief="solid")
comments_text.grid(row=3, column=1, pady=10, padx=20)

# Submit button
submit_button = tk.Button(doctor_frame, text="Υποβολή", bg='green', fg='white',
command=submit_review)
submit_button.grid(row=4, column=1, sticky='e', pady=20)

root.mainloop()

```

Βασικά σημεία κώδικα:

Εισαγωγές Βιβλιοθηκών

```

import tkinter as tk
from tkinter import ttk

```

- **tkinter**: Η κύρια βιβλιοθήκη για τη δημιουργία γραφικών παραθύρων (GUI) στην Python.
- **ttk**: Επέκταση του tkinter που προσφέρει βελτιωμένα widgets.

Συνάρτηση Υποβολής Αξιολόγησης

```

# Function to handle submission
def submit_review():
    rating = rating_var.get()
    comments = comments_text.get("1.0", tk.END).strip()
    print(f"Submitted Rating: {rating}/10")
    print(f"Submitted Comments: {comments}")

```

- **submit_review**: Συνάρτηση που εκτελείται όταν ο χρήστης υποβάλει την αξιολόγησή του. Λαμβάνει την βαθμολογία και τα σχόλια, και τα εκτυπώνει στην κονσόλα.

Ενότητα Αξιολόγησης

```

# Rating section rating_label = tk.Label(doctor_frame, text="Κάνε μια αξιολόγηση:", bg='white',
font=("Helvetica", 12))
rating_label.grid(row=1, column=1, sticky='w', pady=10)
# Stars rating
rating_var = tk.IntVar()

```

```
stars_frame = tk.Frame(doctor_frame, bg='white')
stars_frame.grid(row=2, column=1, sticky='w', pady=10)
for i in range(10):
    star = tk.Radiobutton(stars_frame, text="★", variable=rating_var, value=i+1, bg='white',
fg='green', selectcolor='white', indicatoron=0)
    star.pack(side='left', padx=1)
    rating_scale = tk.Label(stars_frame, text="/10", bg='white', font=("Helvetica", 12))
    rating_scale.pack(side='left', padx=10)
```

- **rating_label:** Ετικέτα που ενημερώνει τον χρήστη να κάνει μια αξιολόγηση.
- **rating_var:** Μεταβλητή για την αποθήκευση της βαθμολογίας.
- **stars_frame:** Πλαίσιο για τα κουμπιά αξιολόγησης με αστέρια (10 Radiobuttons).

Ενότητα Σχολίων

Comments section

```
comments_label = tk.Label(doctor_frame, text="Σχόλια:", bg='white', font=("Helvetica", 12))
comments_label.grid(row=3, column=0, sticky='nw', pady=10)
comments_text = tk.Text(doctor_frame, width=50, height=10, borderwidth=1, relief="solid")
comments_text.grid(row=3, column=1, pady=10, padx=20)
```

- **comments_label:** Ετικέτα για τα σχόλια.
- **comments_text:** Περιοχή κειμένου για την εισαγωγή των σχολίων.

Τέλος Αναφοράς!!!