```java
package application;

import java.time.LocalDate;

public class User<T extends HealthData<?>> {
    private String firstName;
    private String lastName;
    private String email;
    private String password;
    private LocalDate dateOfBirth;
    private String gender;
    private String phoneNumber;
    private ArrayList<T> healthDataList;

    public User(String firstName, String lastName, String email, String password, LocalDate dateOfBirth, String gender, String phoneNumber) {
        if (!isValidEmail(email)) {
            throw new IllegalArgumentException("Invalid email address");
        }

        if (!isValidPassword(password)) {
            throw new IllegalArgumentException("Invalid password");
        }

        if (dateOfBirth.isAfter(LocalDate.now())) {
            throw new IllegalArgumentException("Invalid date of birth");
        }


        this.firstName = firstName;
        this.lastName = lastName;
        this.email = email;
        this.password = password;
        this.dateOfBirth = dateOfBirth;
        this.gender = gender;
        this.phoneNumber = phoneNumber;
        this.healthDataList = new ArrayList<>();
    }

    public void addHealthData(T healthData) {
        healthDataList.add(healthData);
    }

    public void removeHealthData(T healthData) {
        healthDataList.remove(healthData);
    }
    public void editHealthData(int index, T newHealthData) {
        if (index < 0 || index >= healthDataList.size()) {
            throw new IllegalArgumentException("Invalid index for editing health data");
        }

        HealthData<?> existingData = healthDataList.get(index);
        String existingMetric = existingData.getMetric();
        String newMetric = newHealthData.getMetric();

        if (!existingMetric.equals(newMetric)) {
            throw new IllegalArgumentException("Cannot edit health data at index " + index + " with a different metric type");
```

```
61          }
62          System.out.println("Editing health data at index " + index + " for " +
    existingMetric);
63          healthDataList.set(index, newHealthData);
64      }
65
66
67
68
69      public String getFirstName() {
70          return firstName;
71      }
72
73      public void setFirstName(String firstName) {
74          this.firstName = firstName;
75      }
76
77      public String getLastName() {
78          return lastName;
79      }
80
81      public void setLastName(String lastName) {
82          this.lastName = lastName;
83      }
84
85      public String getPassword() {
86          return password;
87      }
88
89      public void setPassword(String password) {
90          this.password = password;
91      }
92
93      public LocalDate getDateOfBirth() {
94          return dateOfBirth;
95      }
96
97      public void setDateOfBirth(LocalDate dateOfBirth) {
98          this.dateOfBirth = dateOfBirth;
99      }
100
101     public String getGender() {
102         return gender;
103     }
104
105     public void setGender(String gender) {
106         this.gender = gender;
107     }
108
109     public void setHealthDataList(ArrayList<T> healthDataList) {
110         this.healthDataList = healthDataList;
111     }
112
113     public ArrayList<T> getHealthDataList() {
114         return healthDataList;
115     }
116
117     public String getFullName() {
118         return firstName + " " + lastName;
```

```
119     }
120
121     public String getEmail() {
122         return email;
123     }
124
125     public void setEmail(String email) {
126         if (!isValidEmail(email)) {
127             throw new IllegalArgumentException("Invalid email address");
128         }
129
130         this.email = email;
131     }
132
133     public String getPhoneNumber() {
134         return phoneNumber;
135     }
136
137     public void setPhoneNumber(String phoneNumber) {
138         this.phoneNumber = phoneNumber;
139     }
140
141     private boolean isValidEmail(String email) {
142         String emailRegex = "^[a-zA-Z0-9_+&*-]+(?:\\." +
143                             "[a-zA-Z0-9_+&*-]+)*@" +
144                             "(?:[a-zA-Z0-9-]+\\.)+[a-z" +
145                             "A-Z]{2,7}$";
146
147         Pattern pattern = Pattern.compile(emailRegex);
148         Matcher matcher = pattern.matcher(email);
149
150         return matcher.matches();
151     }
152
153     private boolean isValidPassword(String password) {
154         // Password must have at least 8 characters, one uppercase letter, one
    lowercase letter, and one digit
155         String passwordRegex = "^(?=.*[0-9])(?=.*[a-z])(?=.*[A-Z]).{8,}$";
156
157         Pattern pattern = Pattern.compile(passwordRegex);
158         Matcher matcher = pattern.matcher(password);
159
160         return matcher.matches();
161     }
162 }
163
```