# NHPlus

**- User-Stories, Akzeptanzkriterien, Testfälle -**

1) Plane innerhalb deiner Gruppe die noch ausstehenden Module, indem du
   a) User-Stories formulierst
   b) über die Formulierung von Akzeptanzkriterien definierst, wann eine User-Story fertig implementiert ist
   **c)** aus der User-Story und ihren Akzeptanzkriterien Tasks ableitest. **Die Tasks sollen genau festlegen, welche Klassen neu zu implementieren und welche Klassen an welcher Stelle um welches Element anzupassen sind.**
   d) abschließend aus den Akzeptanzkriterien für die Testphase Testfälle ableitest.

   Benutze dafür die unten abgebildete Tabellenform. Für weitere User-Stories ist die Tabellenstruktur zu kopieren.

| **User-Story** |
| --- |
| Als Benutzer möchte ich, dass das Programm alle Aspekte der DSGVO abdeckt. Das Aufnehmen des Vermögensstandes der Patienten ist keine wichtige Erhebung von Patientendaten und daher nicht DSGVO konform. Demnach muss die Anwendung dementsprechend angepasst werden. |
| **Akzeptanzkriterien** |
| A_1: Entfernung der Spalte „Vermögensstand“ aus der Anwendung.<br>A_2: Entfernung des Eingabefeldes „Vermögenstand“ aus der Anwendung.<br>A_3: Entfernung des Vermögensstandes aus der Datenbank. |
| **Tasks** |
| T_1: In der View „Vermögensstand“ entfernen.<br>T_2: Entfernung des Eintrages „ASSETS“ aus der Datenbank Tabelle „PATIENT“.<br>T_3: Entfernung des Eintrags „txtAssets“ aus der xml Datei „AllPatientView“.<br>T_4: Alle Verbindungen zu Vermögensstand aus dem Java-Code entfernen.<br>T_5: Entfernung gespeicherter Vermögensstände von bestehenden Patienten. |
| **Testfälle**<br>**TF1_:**<br>- **Vorbedingung**: Das Programm ist gestartet.<br>- **auszuführende Testschritte**: Den Button „Patient/innen“ betätigen.<br>- **erwartetes Ergebnis**: Patiententabelle wird aufgerufen und Spalte „Vermögensstand“ ist nicht sichtbar.<br>**TF2_:** |

- **Vorbedingung**: Das Programm ist gestartet.

- **auszuführende Testschritte**: Den Button „Patient/innen“ betätigen.

- **erwartetes Ergebnis**: Patiententabelle wird aufgerufen und Textfeld „Vermögensstand“ in dem Bereich zur Erstellung neuer Patienten nicht sichtbar.

**TF3_:**

- **Vorbedingung**: Datenbank ist aufgerufen.

- **auszuführende Testschritte**: Konsole der Datenbank aufrufen. Mit Benutzername und Benutzerpasswort mit der Datenbank verbinden. Eine SQL anfrage z.B. (SELECT * FROM Patient) eingeben. Eingegebene SQL anfrage ausführen.

- **erwartetes Ergebnis**: Spalte „Vermögensstand“ in der Tabelle nicht sichtbar.

# NHPlus

**- User-Stories, Akzeptanzkriterien, Testfälle -**

1) Plane innerhalb deiner Gruppe die noch ausstehenden Module, indem du
   a) User-Stories formulierst
   b) über die Formulierung von Akzeptanzkriterien definierst, wann eine User-Story fertig implementiert ist
   **c)** aus der User-Story und ihren Akzeptanzkriterien Tasks ableitest. **Die Tasks sollen genau festlegen, welche Klassen neu zu implementieren und welche Klassen an welcher Stelle um welches Element anzupassen sind.**
   d) abschließend aus den Akzeptanzkriterien für die Testphase Testfälle ableitest.

   Benutze dafür die unten abgebildete Tabellenform. Für weitere User-Stories ist die Tabellenstruktur zu kopieren.

| **User-Story** |
|---|
| Als Administrator möchte ich den Zugriff auf die Anwendung mit einem Accountsystem schützen, um die<br><br>Daten vor Missbrauch durch Dritte zu sichern. |
| **Akzeptanzkriterien** |
| A_1: Bei Start der Anwendung wird nach Benutzername und Kennwort gefragt<br><br>A_2: Bei korrekter Eingabe kann die Anwendung verwendet werden<br><br>A_3: Bei inkorrekter Eingabe wird der Zugriff verweigert und es gibt eine entsprechende Meldung der Anwendung<br><br>A_4: In der Datenbank sollen Nutzername und Passwort der Benutzer gespeichert sein<br><br>A_5: Bei Erstellen eines Benutzers wird ein Passwort vorgegeben, bei erstmaliger Anmeldung muss der Nutzer ein neues Passwort festlegen |
| **Tasks** |
| T_1: In der Datenbank müssen Nutzername+Passwort Kombination für Anwendungsnutzer in einer eigenen Tabelle hinterlegt sein<br><br>T_2: Es muss eine Klasse erstellt werden, die die Anmeldung steuert |
| **Testfälle**<br><br>**TF1_:**<br><br>- **Vorbedingung**:<br><br>- **auszuführende Testschritte**: Anwendung starten -> Anmeldung durchführen<br><br>- **erwartetes Ergebnis**: Anmeldung wird akzeptiert, Anwendung kann genutzt werden |

**TF2_:**

- **Vorbedingung**: TF_1

- **auszuführende Testschritte**: Anwendung starten -> erstmalige Anmeldung durchführen

- **erwartetes Ergebnis**: Anwendung bittet darum ein neues Passwort festzulegen

**TF3_:**

- **Vorbedingung**: TF2_

- **auszuführende Testschritte**: Anwendung starten -> erstmalige Anmeldung durchführen -> vorgeneriertes Passwort als neues Passwort angeben

- **erwartetes Ergebnis**: Anwendung lehnt ab, mit dem Hinweis, dass das neue Passwort nicht mit dem alten identisch sein darf

**TF4_:**

- **Vorbedingung**: TF_1

- **auszuführende Testschritte**: Anwendung starten -> Anmeldung mit falschen Zugangsdaten durchführen

- **erwartetes Ergebnis**: Anwendung lehnt Anmeldung ab, mit dem Hinweis, dass die Anmeldedaten falsch sind

# NHPlus

**- User-Stories, Akzeptanzkriterien, Testfälle -**

1) Plane innerhalb deiner Gruppe die noch ausstehenden Module, indem du
   a) User-Stories formulierst
   b) über die Formulierung von Akzeptanzkriterien definierst, wann eine User-Story fertig implementiert ist
   **c)** aus der User-Story und ihren Akzeptanzkriterien Tasks ableitest. **Die Tasks sollen genau festlegen, welche Klassen neu zu implementieren und welche Klassen an welcher Stelle um welches Element anzupassen sind.**
   d) abschließend aus den Akzeptanzkriterien für die Testphase Testfälle ableitest.

   Benutze dafür die unten abgebildete Tabellenform. Für weitere User-Stories ist die Tabellenstruktur zu kopieren.

| **User-Story** |
| --- |
| Als Administrator möchte ich innerhalb des Accountsystems Zugriffsrechte festlegen können, damit nur die für die eigene Arbeit relevanten Daten bearbeitet werden können. |
| **Akzeptanzkriterien** |
| A_1: Nutzer besitzen, abhängig vom Account, bestimmte Berechtigungen zum Bearbeiten von Daten<br><br>A_2: Der Administrator kann Berechtigungen zuweisen und entziehen |
| **Tasks** |
| T_1: In der Datenbanktabelle für die Benutzeraccounts gibt es einen Eintrag der die Berechtigung festlegt<br><br>T_2: Für verschiedene Berechtigungsgruppe gibt es unterschiedliche Darstellungen der Anwendung<br><br>T_2_1: Folgende Berechtigungsgruppen müssen mindestens vorhanden sein: Administrator, Pfleger, Pflegeassistenz (nur Leseberechtigung)<br><br>T_3: In der Anwendung muss eine Klasse erstellt werden, die anhand der Berechtigungsgruppe den entsprechenden View bereitstellt<br><br>T_4: Es muss eine Übergabe der Berechtigungsgruppe von der Anmeldeklasse an die Berechtigungsklasse erfolgen |
| **Testfälle**<br><br>**TF1_:**<br><br>- **Vorbedingung**: Userstory „Accountsystem“ abgeschlossen<br><br>- **auszuführende Testschritte**: Anwendung starten -> Anmeldung als Administrator durchführen -> Nutzer „Bernd H.“ hinzufügen -> Berechtigungsklasse „Pfleger“ zuweisen |

- **erwartetes Ergebnis**: Ein neuer Account „Bernd H.“ mit den Berechtigungen eines Pflegers wurde angelegt

**TF2_:**

- **Vorbedingung**: Userstory „Accountsystem“ abgeschlossen, TF_1

- **auszuführende Testschritte**: Anwendung starten -> Anmeldung als Pfleger „Bernd H.“ durchführen -> versuchen fremde Behandlung bearbeiten -> eigene Behandlung bearbeiten

- **erwartetes Ergebnis**: Nur die eigene Behandlung kann bearbeitet werden

**TF3_:**

- **Vorbedingung**: Userstory „Accountsystem“ abgeschlossen, TF1_, TF_2

- **auszuführende Testschritte**: Anwendung starten -> Anmeldung als Administrator durchführen -> Nutzer „Bernd H.“ Berechtigungen entziehen -> Nutzer „Bernd H.“ Berechtigungsklasse „Pflegeassistenz“ zuweisen

- **erwartetes Ergebnis**: Der Account Bernd H. werden die Berechtigung als „Pfleger“ entzogen und er bekommt die Berechtigungsklasse „Pflegeassistenz“ zugewiesen

**TF4_:**

- **Vorbedingung**: Userstory „Accountsystem“ abgeschlossen, TF1_-TF3_

- **auszuführende Testschritte**: Anwendung starten -> Anmeldung als Pflegeassistenz „Bernd H.“ durchführen -> Behandlungen anzeigen -> versuchen Behandlung zu bearbeiten -> versuchen eigene Behandlung einzutragen

- **erwartetes Ergebnis**: Behandlungen können lediglich eingesehen, nicht jedoch verändert oder neu angelegt werden

**TF5_:**

- **Vorbedingung**: Userstory „Accountsystem“ abgeschlossen, TF1_-TF4_

- **auszuführende Testschritte**: Anwendung starten -> Anmeldung als Administrator durchführen -> Nutzer „Bernd H.“ Berechtigungen entziehen -> Nutzer „Bernd H.“ löschen

- **erwartetes Ergebnis**: Der Account „Bernd H.“ wurde gelöscht

# NHPlus

**- User-Stories, Akzeptanzkriterien, Testfälle -**

1) Plane innerhalb deiner Gruppe die noch ausstehenden Module, indem du
   a) User-Stories formulierst
   b) über die Formulierung von Akzeptanzkriterien definierst, wann eine User-Story fertig implementiert ist
   **c)** aus der User-Story und ihren Akzeptanzkriterien Tasks ableitest. **Die Tasks sollen genau festlegen, welche Klassen neu zu implementieren und welche Klassen an welcher Stelle um welches Element anzupassen sind.**
   d) abschließend aus den Akzeptanzkriterien für die Testphase Testfälle ableitest.

   Benutze dafür die unten abgebildete Tabellenform. Für weitere User-Stories ist die Tabellenstruktur zu kopieren.

| **User-Story** |
|---|
| Als Schichtleitung möchte ich eine komplette Übersicht der Pfleger, mit ihrem vollen Namen und ihrer<br>Telefonnummer, um sie auf Station erreichen und verwalten zu können. |
| **Akzeptanzkriterien** |
| A_1: Innerhalb des Programms ist eine vollständige Auflistung aller Pflegekräfte aufrufbar<br>A_2: Innerhalb dieser Auflistung soll die Telefonnummer ersichtlich sein<br>A_3: Die Daten werden in der Datenbank gespeichert und von dort aufgerufen<br>A_4: Neue Pflegekräfte können innerhalb der Anwendung eingetragen werden<br>A_5: Bestehende Pflegekräfte können entfernt oder gesperrt werden<br>A_6: Daten bestehender Pflegekräfte können verändert werden |
| **Tasks** |
| T_1: Es muss eine CaregiverDAO-Klasse erstellt werden<br>T_2: Es muss eine Caregiver Model-Klasse erstellt werden<br>T_3: Es muss eine neue Tabelle in der Datenbank initialisiert werden |
| **Testfälle**<br>**TF1_:**<br>**- Vorbedingung:**<br>**- auszuführende Testschritte:** Programm öffnen -> Knopf „Pfleger/innen“ drücken<br>**- erwartetes Ergebnis:** Auflistung von Pflegern mit vollständigem Namen und Telefonnummer öffnet sich |

**TF2_:**

- **Vorbedingung:** TF1 erfolgreich

- **auszuführende Testschritte:** TF1 -> Neuen Pfleger hinzufügen

- **erwartetes Ergebnis:** Neuer Pflegereintrag in Anwendung und DB

**TF3_:**

- **Vorbedingung:** TF1 erfolgreich

- **auszuführende Testschritte:** TF1 -> Bestehende Pfleger löschen/sperren

- **erwartetes Ergebnis:** Pfleger wird gelöscht/aus der Anwendung entfernt

**TF4_:**

- **Vorbedingung:** TF1 erfolgreich

- **auszuführende Testschritte: TF1 ->** Telefonnummer eines bestehenden Pflegers modifizieren

- **erwartetes Ergebnis:** Bestehende Daten des Pflegers wurde verändert

# NHPlus

**- User-Stories, Akzeptanzkriterien, Testfälle -**

1) Plane innerhalb deiner Gruppe die noch ausstehenden Module, indem du
   a) User-Stories formulierst
   b) über die Formulierung von Akzeptanzkriterien definierst, wann eine User-Story fertig implementiert ist
   **c)** aus der User-Story und ihren Akzeptanzkriterien Tasks ableitest. **Die Tasks sollen genau festlegen, welche Klassen neu zu implementieren und welche Klassen an welcher Stelle um welches Element anzupassen sind.**
   d) abschließend aus den Akzeptanzkriterien für die Testphase Testfälle ableitest.

   Benutze dafür die unten abgebildete Tabellenform. Für weitere User-Stories ist die Tabellenstruktur zu kopieren.

| **User-Story** |
|---|
| Als Benutzer möchte ich, dass das Programm automatisiert ist und mir hilft den Arbeitsaufwand zu minimieren. Um diesen erfolgreich zu minimieren, soll das Löschen der Patientendaten nach 10 Jahren automatisch geschehen. |
| **Akzeptanzkriterien**<br><br>A_1: Daten werden nach 10 Jahren automatisch gelöscht.<br><br>A_2: Nach Schadensersatzgeltung/Klage wird der Eintrag erst nach 30 Jahren automatisch gelöscht. |
| **Tasks**<br><br>T_1: Implementierung einer automatischen Löschung, welche nach 10 Jahren der Erstellung des Eintrags erfolgt.<br><br>T_2: Angelegte Datensätze werden automatisch mit dem Datum vom Erstellungsdatum versehen.<br><br>T_3: Neue Java Klasse „DeleteDateCheck“ erstellen, welche nach dem Start des Programmes initialisiert wird und alte Daten löscht.<br><br>T_4: Admin hat rechte ein Wahrheitswert bei den Einträgen zu setzen, welche die Datensätze erst nach 30 Jahren nach der Erstellung der Einträge löscht. |
| **Testfälle**<br><br>**TF1_:**<br><br>- **Vorbedingung**: Das Programm ist gestartet.<br><br>- **auszuführende Testschritte**: Den Button „Patient/innen“ betätigen.<br><br>- **erwartetes Ergebnis**: Einträge die älter als 10 Jahre sind wurden gelöscht.<br><br>**TF2_:** |

- **Vorbedingung**: Das Programm ist gestartet.

- **auszuführende Testschritte**: Den Button „Patient/innen“ betätigen.

- **erwartetes Ergebnis**: Einträge, welche älter als 10 Jahre sind, bei denen der Wahrheitswert gesetzt worden ist, sind sichtbar.

**TF3_:**

- **Vorbedingung**: Das Programm ist gestartet.

- **auszuführende Testschritte**: Den Button „Patient/innen“ betätigen.

- **erwartetes Ergebnis**: Einträge, welche älter als 30 Jahre sind, bei denen der Wahrheitswert gesetzt worden ist, sind nicht sichtbar.

# NHPlus

**- User-Stories, Akzeptanzkriterien, Testfälle -**

1) Plane innerhalb deiner Gruppe die noch ausstehenden Module, indem du
   a) User-Stories formulierst
   b) über die Formulierung von Akzeptanzkriterien definierst, wann eine User-Story fertig implementiert ist
   **c)** aus der User-Story und ihren Akzeptanzkriterien Tasks ableitest. **Die Tasks sollen genau festlegen, welche Klassen neu zu implementieren und welche Klassen an welcher Stelle um welches Element anzupassen sind.**
   d) abschließend aus den Akzeptanzkriterien für die Testphase Testfälle ableitest.

   Benutze dafür die unten abgebildete Tabellenform. Für weitere User-Stories ist die Tabellenstruktur zu kopieren.

| **User-Story** |
| --- |
| Als Administrator möchte ich, dass das Programm nicht leicht manipuliert werden kann. Das Löschen durch User, welche keine Admin Berechtigungen besitzen ist eine Sicherheitslücke, welche behoben werden muss. |
| **Akzeptanzkriterien** |
| A_1: Entfernung des „löschen“ Button bei Usern ohne Admin Berechtigungen. |
| **Tasks** |
| T_1: Entfernung des Eintrags „btnDelete“ aus der xml Datei „AllTreatmentView“ für Anwender ohne Adminrechte.<br><br>T_2: Entfernung des Eintrags „btnDelete“ aus der xml Datei „AllPatientView“ für Anwender ohne Adminrechte.<br><br>T_3: Entfernung des Eintrags „btnDelete“ aus der java Datei „AllPatientController“ für Anwender ohne Adminrechte.<br><br>T_4: Entfernung des Eintrags „btnDelete“ aus der java Datei „AllTreatmentController“ für Anwender ohne Adminrechte. |
| **Testfälle**<br><br>**TF1_:**<br><br>- **Vorbedingung**: Das Programm ist gestartet. User besitzt keine Adminrechte.<br><br>- **auszuführende Testschritte**: Den Button „Patient/innen“ betätigen.<br><br>- **erwartetes Ergebnis**: Button „Löschen“ nicht sichtbar.<br><br>**TF2_:** |

- **Vorbedingung**: Das Programm ist gestartet. User besitzt  Adminrechte.

- **auszuführende Testschritte**:  Den Button „Patient/innen“ betätigen.

- **erwartetes Ergebnis**:  Button „Löschen“ sichtbar.

**TF3_:**

- **Vorbedingung**:  Das Programm ist gestartet. User besitzt keine Adminrechte.

- **auszuführende Testschritte**: Den Button „Behandlungen“ betätigen.

- **erwartetes Ergebnis**: Button „Löschen“ nicht sichtbar.

**TF4_:**

- **Vorbedingung**:  Das Programm ist gestartet. User besitzt  Adminrechte.

- **auszuführende Testschritte**: Den Button „Behandlungen“ betätigen.

- **erwartetes Ergebnis**:  Button „Löschen“ sichtbar.

# NHPlus

**- User-Stories, Akzeptanzkriterien, Testfälle -**

1) Plane innerhalb deiner Gruppe die noch ausstehenden Module, indem du
   a) User-Stories formulierst
   b) über die Formulierung von Akzeptanzkriterien definierst, wann eine User-Story fertig implementiert ist
   **c)** aus der User-Story und ihren Akzeptanzkriterien Tasks ableitest. **Die Tasks sollen genau festlegen, welche Klassen neu zu implementieren und welche Klassen an welcher Stelle um welches Element anzupassen sind.**
   d) abschließend aus den Akzeptanzkriterien für die Testphase Testfälle ableitest.

   Benutze dafür die unten abgebildete Tabellenform. Für weitere User-Stories ist die Tabellenstruktur zu kopieren.

| **User-Story** |
|---|
| Als Administrator möchte ich, dass das Programm nicht leicht zu manipulieren ist. Daher muss ein Sperren Button für Patienten- und Behandlungsdaten implementiert werden. |
| **Akzeptanzkriterien** |
| A_1: Möglichkeit der „Sperrung“ von Daten durch berechtigtes Personal. |
| **Tasks** |
| T_1: Implementierung eines „Sperrung“ Buttons bei „Patienten/innen“.<br><br>T_2: Implementierung der „Sperrung“ von Daten im Java-Code.<br><br>T_3: Implementierung der „Sperrung“ von Daten im xml Code.<br><br>T_4: Implementierung eines „Sperrung“ Buttons bei „Behandlungen“. |
| **Testfälle**<br><br>**TF1_:**<br><br>- **Vorbedingung**: Das Programm ist gestartet. User ist entweder Pfleger oder Admin.<br><br>- **auszuführende Testschritte**: Den Button „Behandlungen“ betätigen.<br><br>- **erwartetes Ergebnis**: Button „Sperren“ sichtbar.<br><br>**TF2_:**<br><br>- **Vorbedingung**: Das Programm ist gestartet. User ist entweder Pfleger oder Admin.<br><br>- **auszuführende Testschritte**: Den Button „Patient/innen“ betätigen.<br><br>- **erwartetes Ergebnis**: Button „Sperren“ sichtbar.<br><br>**TF3_:** |

- **Vorbedingung**: Das Programm ist gestartet. User ist entweder Pfleger oder Admin.

- **auszuführende Testschritte**: Den Button „Behandlungen“ betätigen. Anwählen eines Eintrags und drücken des „Sperren“ Buttons.

- **erwartetes Ergebnis**: Eintrag ist nicht mehr sichtbar.

**TF4_:**

- **Vorbedingung**: Das Programm ist gestartet. User ist entweder Pfleger oder Admin.

- **auszuführende Testschritte**: Den Button „Patient/innen“ betätigen. Anwählen eines Eintrags und drücken des „Sperren“ Buttons.

- **erwartetes Ergebnis**: Eintrag ist nicht mehr sichtbar.