Anhang A: Product Backlog

ID	User Story	Priorität	Akzeptanzkriterien
PB-1	Als Mitarbeiter*in möchte ich neue Patientendaten erfassen, damit diese im System verfügbar sind.	Hoch	Patient kann mit Nachname, Vorname, Geburtsdatum, Adresse, Kontakt (Telefonnummer <i>oder</i> Email) und Krankenkasse angelegt werden; Daten erscheinen in der Patientenliste.
PB-2	Als Mitarbeiter*in möchte ich bestehende Patientendaten bearbeiten können, um Änderungen (z. B. Adresse, Kontakt) nachzupflegen.	Hoch	Auswahl eines Patienten → Bearbeitungsmaske → Änderungen werden gespeichert und angezeigt.
PB-3	Als Mitarbeiter*in möchte ich Patienten löschen, damit Karteileichen entfernt werden können.	Mittel	Patient lässt sich auswählen und endgültig entfernen.
PB-4	Als Mitarbeiter*in möchte ich Termine für Patienten anlegen, um deren Praxisbesuche zu verwalten.	Hoch	Termin kann mit PatientID, Datum/Uhrzeit und Grund gespeichert werden.
PB-5	Als Mitarbeiter*in möchte ich Termine bearbeiten können, um Änderungen flexibel zu berücksichtigen.	Hoch	Termin lässt sich nach Auswahl editieren und speichern; Änderungen erscheinen sofort in der Liste.
PB-6	Als Mitarbeiter*in möchte ich Termine löschen, damit falsche oder abgesagte Termine entfernt werden können.	Mittel	Termin kann ausgewählt und endgültig entfernt werden.
PB-7	Als Mitarbeiter*in möchte ich eine Suchfunktion nach Patienten haben, um gezielt bestimmte Einträge zu finden.	Mittel	Eingabe von Nachname → gefilterte Anzeige in Tabelle.
PB-8	Als Mitarbeiter*in möchte ich eine Suchfunktion nach Terminen haben, um bestimmte Termine (z. B. nach Datum) schneller zu finden.	Mittel	Eingabe von Datum \rightarrow gefilterte Anzeige in Tabelle.
PB-9	Als Mitarbeiter*in möchte ich, dass das GUI übersichtlich und bedienbar ist, damit die Anwendung im Praxisalltag praktikabel ist.	Hoch	Übersichtliche Tabellen für Patienten und Termine, Schaltflächen für CRUD- Operationen, Suchmasken.
PB-10	Als Entwickler möchte ich automatisierte Tests schreiben, um die korrekte Funktionsweise der Kernlogik sicherzustellen.	Niedrig	JUnit-Tests für Patienten- und Terminverwaltung laufen erfolgreich durch.

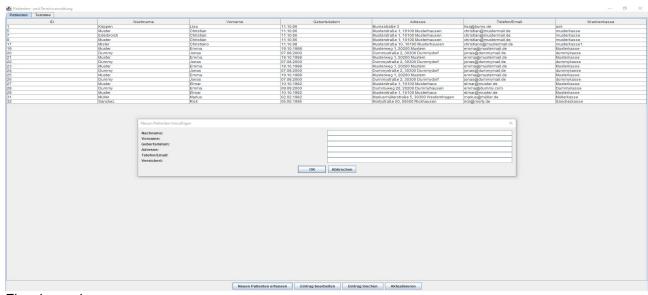
Anhang B: Testdokumentation

Manuelle Tests:

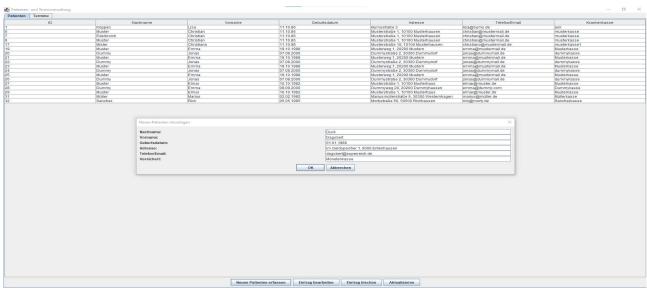
Test-ID	Vorgang	Eingabe	Ergebnis - Soll	Ergebnis - Ist	Status
MT01	erstellePatient()	"Sanchez", "Rick", "05.05.1995", "Mortystraße 50, 50500 Rickhausen", "rick@morty.de", "Sanchezkasse"	Neuer Patient erstellt und erscheint in Liste	Neuer Patient mit allen Daten der Liste hinzugefügt	bestanden
MT02	erstelleTermin()	Eingabe PatientID 31 über Dropdown, dann "10.10.2025, 11:30 Uhr", "Heiserkeit"	Neuer Termin erstellt, erscheint in Liste	Neuer Termin mit allen Daten hinzugefügt	Fehler , behoben in Sprint 3
MT03	loeschePatient()	Auswahl Patient in Liste, dann "Eintrag löschen"	Eintrag wird entfernt	Eintrag gelöscht	bestanden
MT04	bearbeitePatient()	Auswahl Patient in Liste, dann "Eintrag bearbeiten", neu: "Musterstraße 99, 90900 Musterdorf", dann "Ok"	Termin wird entfernt	Termin gelöscht	bestanden
MT05	zeigeAllePatienten()	bei Aufruf der Anwendung/bei "Aktualisieren" nach Suche	Liste mit Patienten wird angezeigt	Alle Patienten aufgelistet	bestanden
МТ06	zeigeAlleTermine()	bei Aufruf der Anwendung/bei "Aktualisieren" nach Suche	Liste mit Terminen wird angezeigt	Alle Termine aufgelistet	Fehler "no such column ,terminld"", behoben in Sprint 3

Automatisierte Tests:

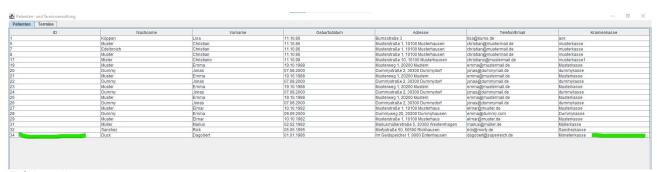
Erfassen eines neuen Patienten:



Eingabemaske

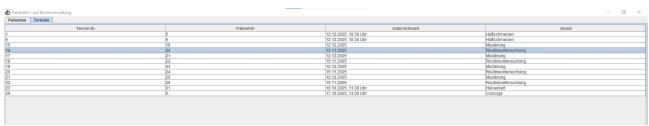


Eingabe der Daten

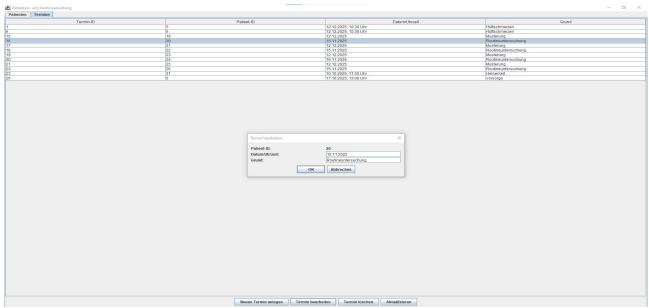


Erfolgreich angelegt

Termin bearbeiten:



Auswahl des Termins

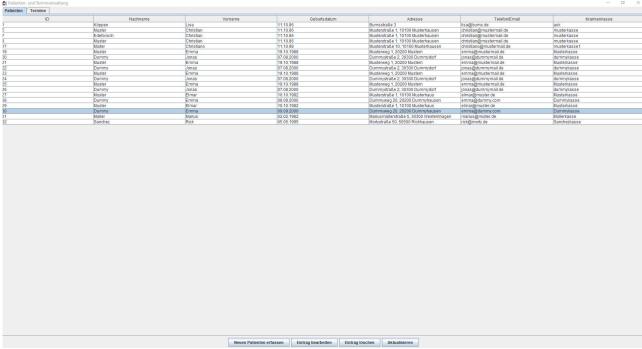


Eingabe neuer Daten



Termin geändert

Patient löschen:

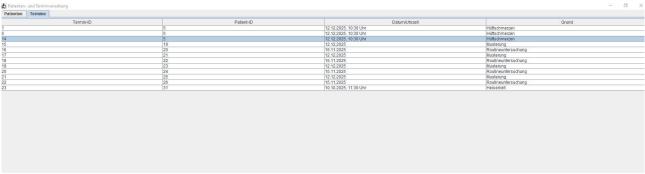


Auswahl des Patienten

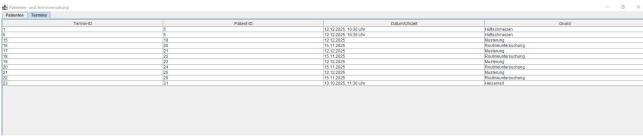


Eintrag entfernt

Termin löschen:



Termin auswählen



Eintrag entfernt

Anhang C: GitHub-Repository

https://github.com/AnKi-4242/Agile-Softwareentwicklung---Projekt