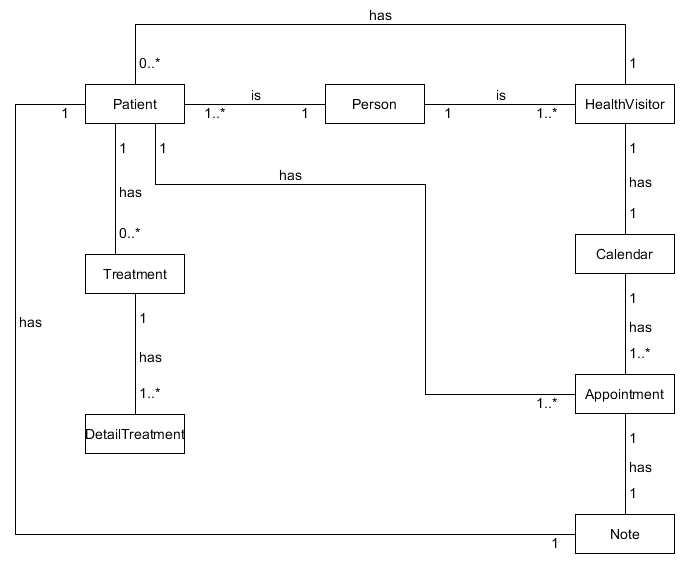
Domain Model and Sequence Diagram

Application design

Erstellt durch Gruppe Blau

# UML domainmodel

Um die Funktionen unserer App sicherzustellen, benötigen wir sicher folgende Klassen, mit den ersichtlichen Abhängigkeiten.



## CRC-Cards

Anhand des Domain Models erstellten wir folgende CRC-Cards:

|  |  |
| --- | --- |
| **Class Name:** Person | |
| **Purpose:** Beinhaltet die Grundattribute für Sozialarbeiter und Patient. | |
| **Responsibilities:**   * Kennt Namen * Kennt Vornamen * Kennt Adresse * Kennt Geschlecht * Kennt Telefon Nr. * Kennt Geburtsdatum | **Collaborators:**   * Wird von HealthVisitor und Patient benutzt * Benutz Address Klasse zur genaueren Definition |

|  |  |
| --- | --- |
| **Class Name:** Patient | |
| **Purpose:** Speichert alle Patientenrelevanten Informationen | |
| **Responsibilities:**   * Kennt alle Appointments * Kennt Treatment * Kennt HealthVisitor * Kennt Pat-ID * Kennt Kommentar | **Collaborators:**   * Benutzt Treatment um den Behandlungsverlauf abzubilden * Subklasse von Person * Benutzt Appointment zum Darstellen seiner Termine * Gehört zu einem HealthVisitor |

|  |  |
| --- | --- |
| **Class Name:** HealthVisitor | |
| **Purpose:** Speichert alle Daten für den Sozialarbeiter | |
| **Responsibilities:**   * Kennt seinen Calendar * Kennt seine Institution * Kennt alle seine Patient | **Collaborators:**   * Benutzt Calendar um seine Termine zu verwalten * Benutzt Patient um seine Patienten zu verwalten * Ist Subklasse von Person |

|  |  |
| --- | --- |
| **Class Name:** Calendar | |
| **Purpose:** Verwaltet Termine eines Sozialarbeiters | |
| **Responsibilities:**   * Kennt HealthVisitor * Kennt Appointment | **Collaborators:**   * Gehört zu HealthVisitor * Benutzt Appointment zum Darstellen der Termine |

|  |  |
| --- | --- |
| **Class Name:** Appointment | |
| **Purpose:** Beinhaltet die Eigenschaften eines Termins | |
| **Responsibilities:**   * Kennt Anfangszeit * Kennt Endzeit * Kennt Ort * Kennt Datum * Kennt Note * Kennt Patient | **Collaborators:**   * Gehört zu Calendar * Benutzt Patient um Abhängigkeit darzustellen * Benutz Note um zusätzliche Informationen zu Speichern |

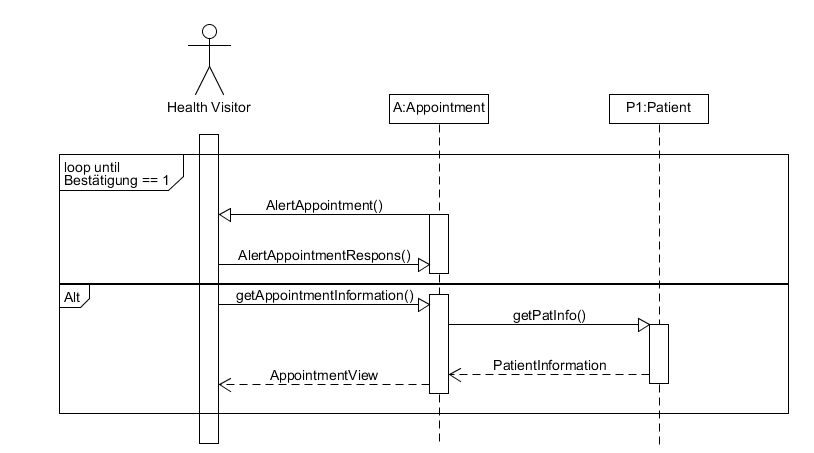
|  |  |
| --- | --- |
| **Class Name:** Note | |
| **Purpose:** Beinhaltet Freitext um zusätzliche Informationen einer Klasse hinzuzufügen | |
| **Responsibilities:**   * Kennt Appointment * Kennt Patient | **Collaborators:**   * Gehört zu Appointment * Gehört zu Patient |

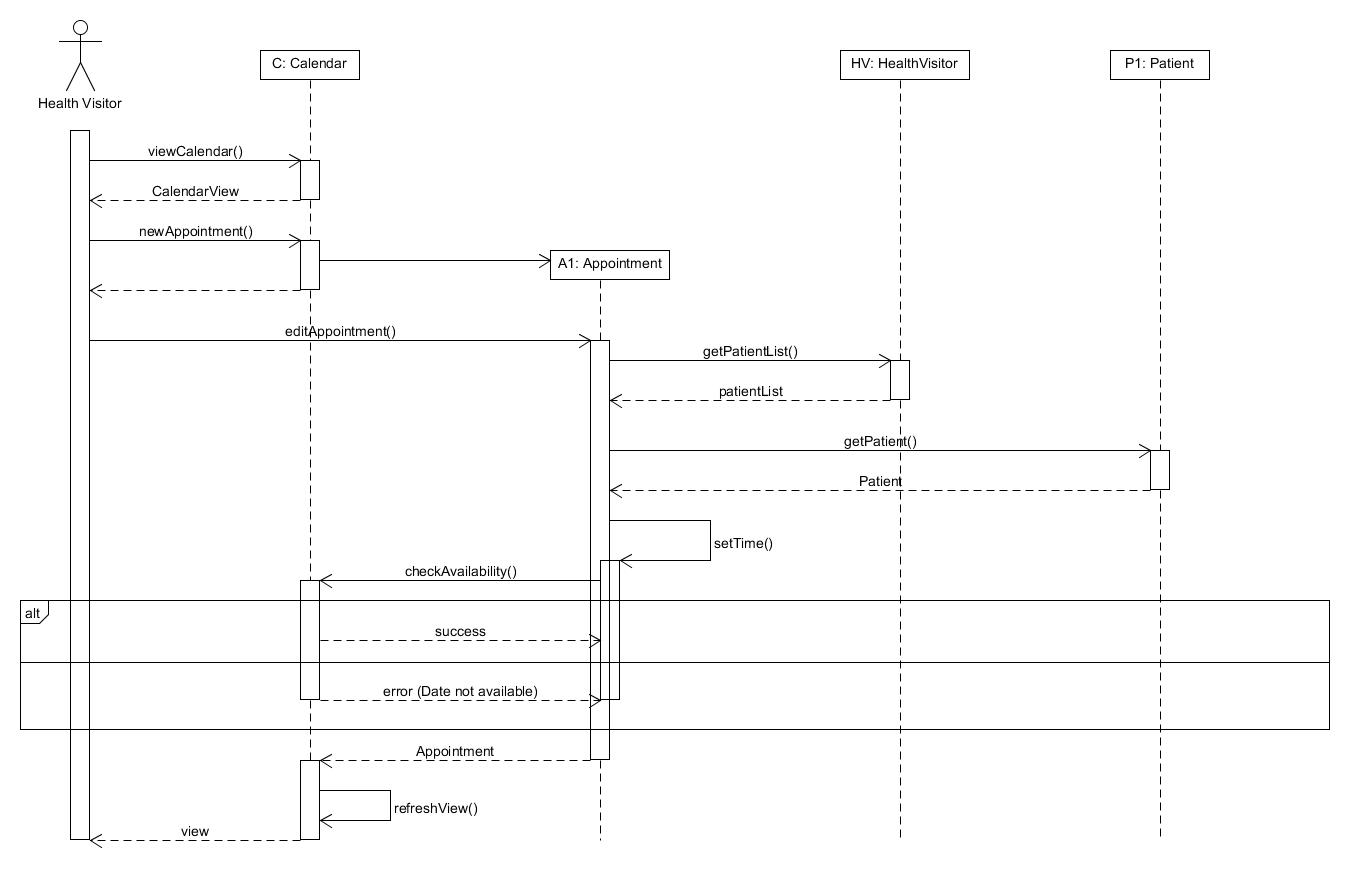
|  |  |
| --- | --- |
| **Class Name:** Treatment | |
| **Purpose:** Speichert alle behandlungsrelevanten Informationen eines Patienten ab. | |
| **Responsibilities:**   * Kennt Patient * Kennt DetailTreatment * Kennt Eintrittsdatum * Kennt Austrittsdatum * Kennt Klinikinfo | **Collaborators:**   * Gehört zu Patient * Benutzt DetailTreatment um eine genaue Behandlungsübersicht darzustellen |

|  |  |
| --- | --- |
| **Class Name:** DetailTreatment | |
| **Purpose:** Speichert alle Detailinformationen zu einer Behandlung | |
| **Responsibilities:**   * Kennt Behandlungsgrund * Kennt behandelnden Arzt * Kennt Datum * Kennt Treatment | **Collaborators:**   * Gehört zu Treatment |

# Sequencediagram

Anhand der CRC-Cards wie auch dem Domain Model, erstellten wir folgende zwei Sequenzdiagramme für die Prozesse „Alert bei einem Termin“ und „Calender Update and Create newAppointment“.





# Erweitertes Domain Model

Durch unsere Sequenzdiagramme kann das vorhandene Domain Model wie folgt angepasst werden (zu einem pseudo Class Diagram).

