Ziele:

# Google Schnittstelle:

Die Google Schnittstelle soll es einem Ermöglichen eine Verbindung zur Google Kalender API aufzubauen und an die verschiedene Request schicken zu können. Einerseits das Erstellen von Terminen aus seinen Google Konto außerdem das Löschen von Terminen auf seinem Konto und das bearbeiten seiner Termine auf seinem Konto. Des weiterem soll es nach Möglichkeit auch bestehende Termine aus CIS auf sein Konto zu synchronisieren. Da die App auch im Offlinemodus funktionieren soll werden Anfragen an die API periodisch abgewickelt und nur im Online Modus. Optional: Synchronisierung wahlweiße nur mit WLAN nicht mit mobilen Internet.

# User Interface:

Eines unserer Ziele ist es die Möglichkeit zu bieten unser Produkt auch zu verwenden. Dafür brauchen wir eine Benutzeroberfläche. Die wollen wir stark an die Google Kalender Oberfläche anlehnen.  
Optional: Verschiedene Themen zu implementieren.

# Termine editieren/erstellen/löschen:

Wie bereits erwähnt sollen Termine bearbeitet, erstellt und löschbar sein. Da der Mehrwert unserer App in der dynamischen Termin Verteilung steckt wir hier auch ein großer Teil unserer Aufmerksamkeit sein. Siehe unten. Diese Funktionen sollen sowohl Online als auch Offline verfügbar sein. Siehe „Google Schnittstelle“. Es sollte möglich sein einen Termin mit seinen Unterterminen zusammen löschen zu können und außerdem seine Verteilung von Unterterminen auch nachträglich zu ändern.

# Dynamische Terminverteilung:

Die dynamische Terminverteilung ist das Kernstück unserer App. Hier soll es möglich sein bei der Erstellung eines Termines ihn unterteilen zu können und Untertermine entweder automatisch oder mithilfe von vorgeschlagenen Daten und Uhrzeiten händisch zu verteilen. Dafür ein Beispiel: Johannes hat in einen Monat einen Test und möchte sich seine Lernzeiten aufschreiben um zu planen. Er Tippt den Termin des Tests ein und wählt die Option einen Multi Date Termin daraus zu machen. Diese Unterpakete kann er jetzt benennen, er entscheidet sich für den Namen „Für Test lernen“. Johannes bekommt eine Vorschau wie die App jetzt dynamisch seine Termine verteilen würde. Dabei hat er verschiedene Optionen wie er diese Termine verteilt haben möchte und in wie große Blöcke er sie aufspaltet. Er entscheidet sich dafür progressiv mehr lernen zu wollen und speichert den Termin ab.

# Lokaler & Online Modus:

Wie bei der Google Schnittstelle schon erklärt wird das Programm obwohl es viel mit dem Google Kalender arbeitet auch Offline verfügbar sein. Dafür werden wir eine Datenstruktur lokal auf dem Handy haben. Die Serialisierung von Objekten werden wir mit einer externen Library abdecken.

Nicht Ziele:

# Kalender ganz neu ausprogrammieren:

Unser Kalender wird von design stark an den Google Kalender anlehnen. Abgesehen davon werden wir schon ausprogrammierte Kalenderteile aus dem Android Studio nehmen und den Kalender nicht von Grund auf neu schreiben.

# Responsiv Design:

Wir werden nicht auf verschiedene Arten von Bildschirmauflösungen eingehen oder auf die Möglichkeit sein Smartphone drehen zu können.

# Kontoverwaltung:

Die Preferencen eines Users werden zwar auslagerbar sein und es wird möglich sein sie auch wieder einzulesen. Aber eine Online Kontoverwaltung mit Password und Usern wird es nicht geben.

# Datenbank:

Da wir mit der Google API arbeiten sehen wir keinen Sinn darin eine Datenbank für unser Projekt zu verwenden. Alle Daten liegen entweder serialisiert bis zu einem gewissen Ausmaß auf dem Smartphone oder Online auf dem Google Kalender.

# Coole Effekte:

Es werden außer Standartanimationen keine weiteren Animationen in unser Programm eingebaut.

Spezifikationen:

# Google API Klasse:

Funktion siehe „Ziele: Google API“

# Kalenderklasse:

Die Kalenderklasse wird die Logik hinter den dynamischen Terminverteilen beinhalten. Außerdem werden hier verschiedene Filteroptionen ausprogrammiert und die Fähigkeit nach bestimmten Terminen zu suchen. Die Kalenderklasse wird außerdem Funktionen beinhalten die einen das Löschen und Bearbeiten von Terminen ermöglicht.

# Eintrag:

Ein Eintrag ist eine allgemein gehaltene Überklasse für Single Date Eintrag und Multi Date Eintrag. Hierbei wird es sich entweder um eine Klasse oder eine Schnittstelle handeln. Das wird sich beim Programmieren dann Zeigen. Sie wird Basisfunktionen beinhalten wie zum Beispiel show\_Eintrag().

# Single Date Eintrag:

Eine Klasse die einen klassischen Termin beschreibt und Funktionalitäten ihrer Parentklasse zur Verfügung stellt. Hier werden Informationen gespeichert wie das Start und End-Datum, die Dauer, der Ort und die Priorität sowie die Kategorie. Die letzten 2 Parameter sind jeweils Filteroptionen und Informationen für die dynamische Verteilung.

# Multi Date Eintrag:

Ein Multi Date Eintrag speichert einerseits den Single Date Eintrag der ursprünglich erstellt wurde und andererseits eine Liste von Single Date Einträgen die zusätzlich beim aufsplitten des Termins entstanden sind.

# File Manager:

Der File Manager kümmert sich um alles was mit Serialisierung zu tun hat. Zugriffe auf Daten auf dem Handy passieren nur über diese Klasse.

# Userpräferenzen:

Das ist eine Klasse rein zum Speichern der Präferenzen eines Users sowie seinen Namen für personalisierte Pop-Ups und Nachrichten.