

Figure 1. Structure of the UI Component. The Class Diagram shows how the **UI** components interact with each other.

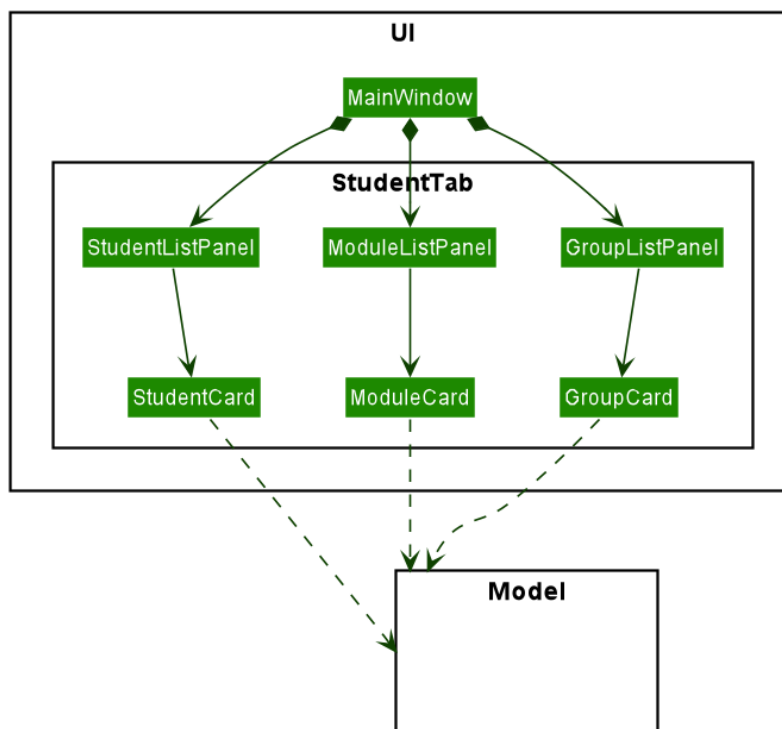


Figure 2. Structure of the Student Tab Component. The Class Diagram shows how the components in the **Student Tab** interact with each other.

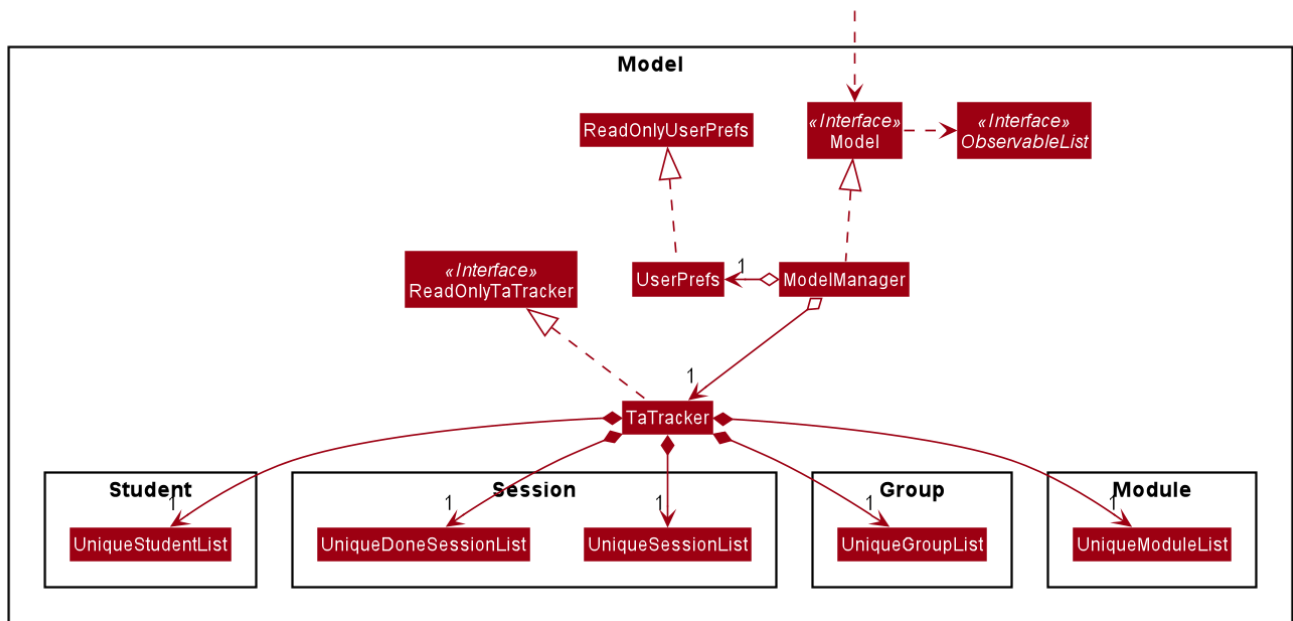


Figure 3. Structure of the Model Component. The Class Diagram shows how the different Model components interact with each other.

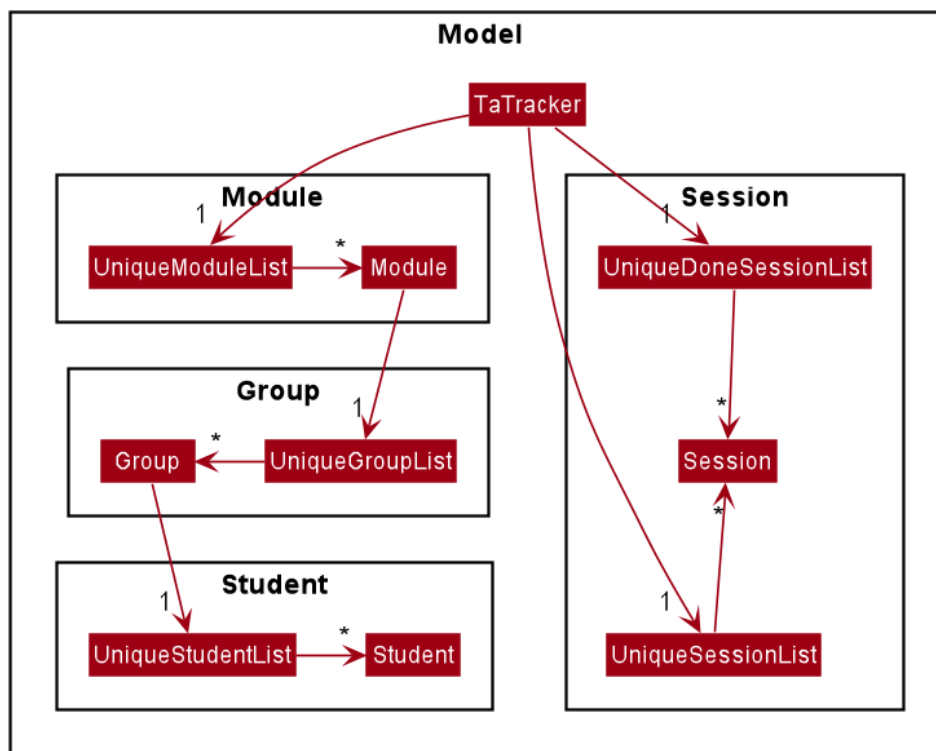


Figure 4. Model Components - Class Diagram. The Class Diagram shows the relationship between the different classes in the Model component.

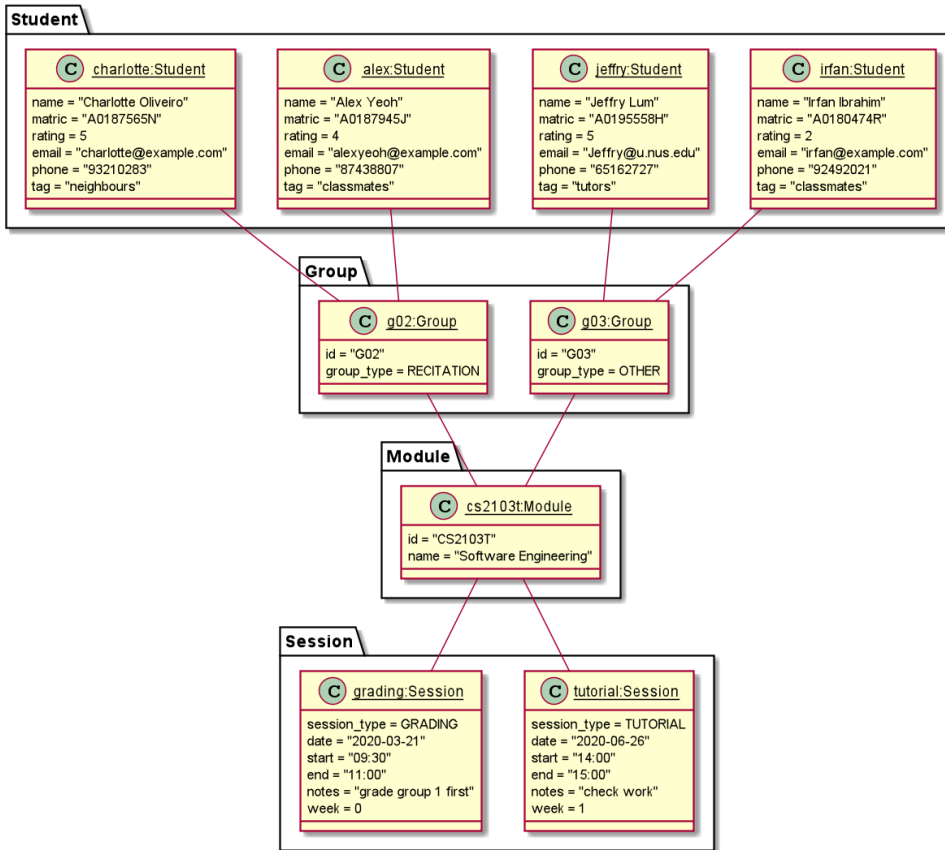


Figure 5. Model Components - Object Diagram. The Object Diagram shows an example of the relationship between the different Model objects. This example is based on the state of TA-Tracker when it is first run (without any user data).

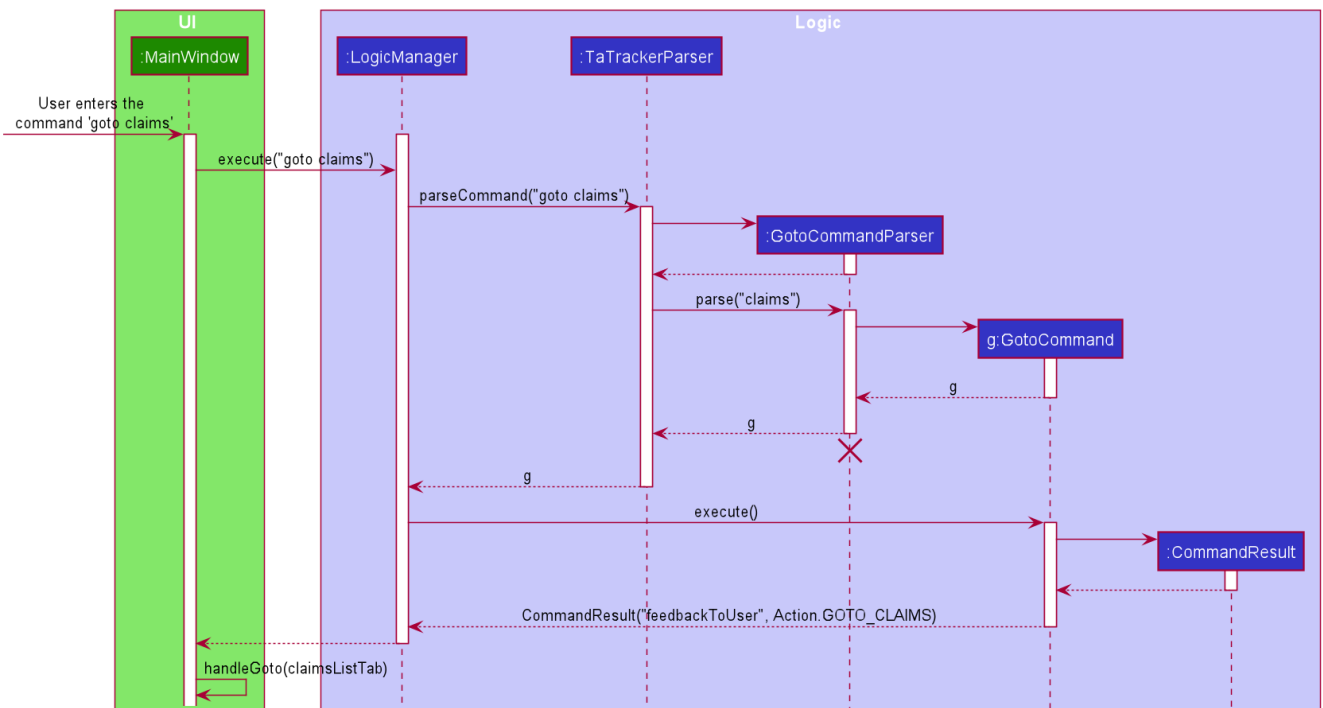
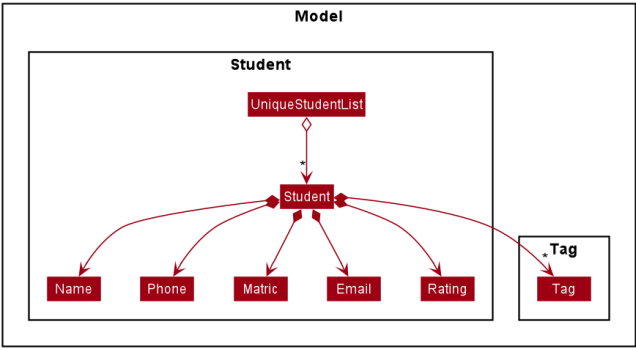
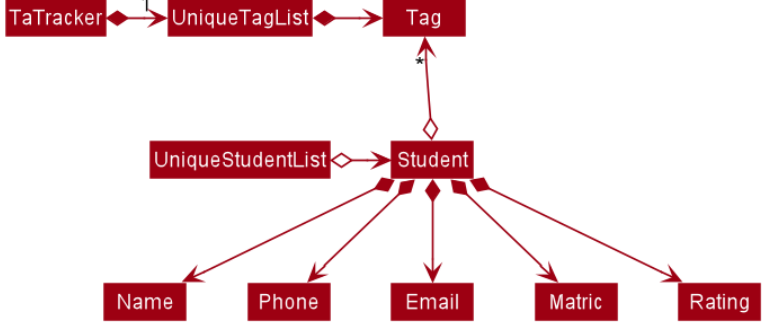
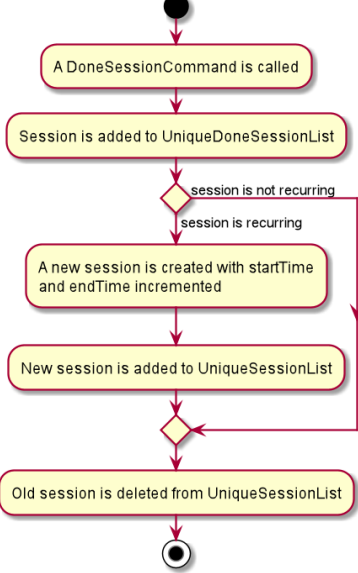
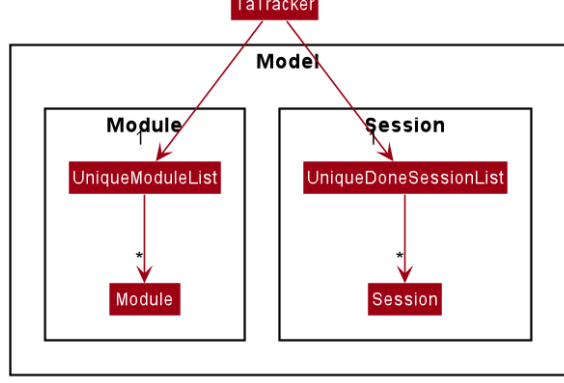


Figure 6. Sequence Diagram for Goto Claims Command. The Sequence Diagram shows the interactions between the Logic and UI components of the TA-Tracker when the user enters the command goto claims.

Description	Diagram
<p>Structure of the Student Component. The <i>Class Diagram</i> shows how different classes are related in the functioning of a Student Object.</p>	 <pre> classDiagram class Model { class Student { UniqueStudentList Student Name Phone Matric Email Rating } class Tag { Tag } UniqueStudentList "*" -- "*" Student Student --> Name Student --> Phone Student --> Matric Student --> Email Student --> Rating Student --> "*" Tag Tag --> "*" Tag </pre>
<p>As a more OOP model, we can store a Tag list in TaTracker, which Student can reference. This is an example of what such a model may look like.</p>	 <pre> classDiagram class TaTracker { UniqueTagList } class UniqueTagList { Tag } class Tag { } class UniqueStudentList { Student } class Student { Name Phone Email Matric Rating } TaTracker --> "1" UniqueTagList UniqueTagList --> "*" Tag UniqueStudentList --> "*" Student Student --> Name Student --> Phone Student --> Email Student --> Matric Student --> Rating </pre>
<p>Session Done- Activity Diagram. The following <i>Activity Diagram</i> describes how TaTracker is updated when a SessionDone command is entered.</p>	 <pre> graph TD Start(()) --> A[A DoneSessionCommand is called] A --> B[Session is added to UniqueDoneSessionList] B --> C{ } C -- "session is not recurring" --> D[A new session is created with startTime and endTime incremented] C -- "session is recurring" --> E[New session is added to UniqueSessionList] D --> E E --> F{ } F --> G[Old session is deleted from UniqueSessionList] G --> End((())) </pre>
<p>Claims View - Class Diagram. The following <i>Class Diagram</i> shows how different classes are related in the functioning of the Claims View.</p>	 <pre> classDiagram class TaTracker { Module Session } class Model { class Module { UniqueModuleList Module } class Session { UniqueDoneSessionList Session } UniqueModuleList --> "*" Module UniqueDoneSessionList --> "*" Session } TaTracker --> Module TaTracker --> Session </pre>