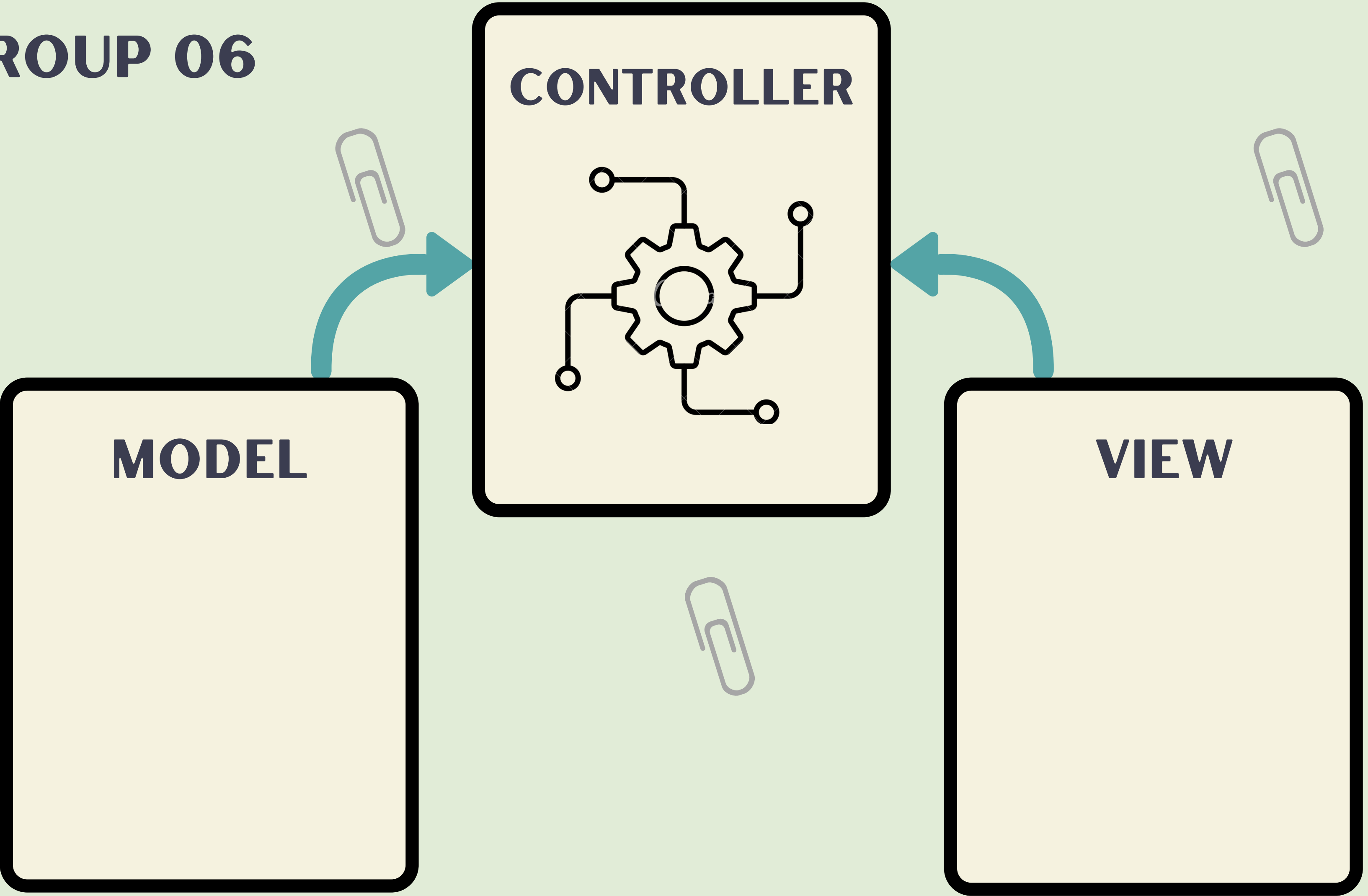# GROUP 06

CONTROLLER

MODEL
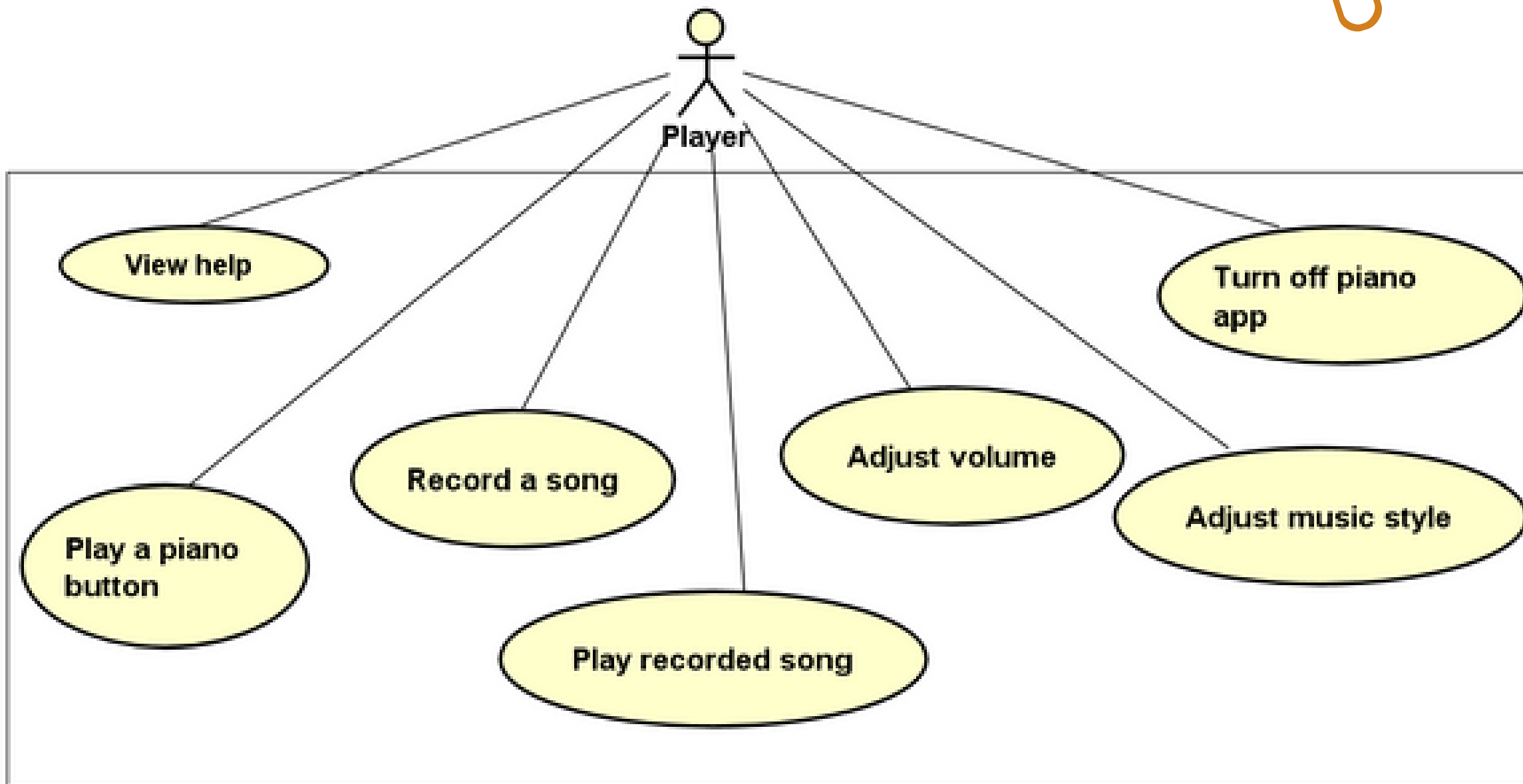
VIEW

PROBLEM STATEMENT

PROBLEM

Implement an application that provides GUI for the user to virtually play an electronic piano.

DESIGN

Design a UI with keyboard, volume, record, music style, ... that comfort users.

USE CASE DIAGRAM

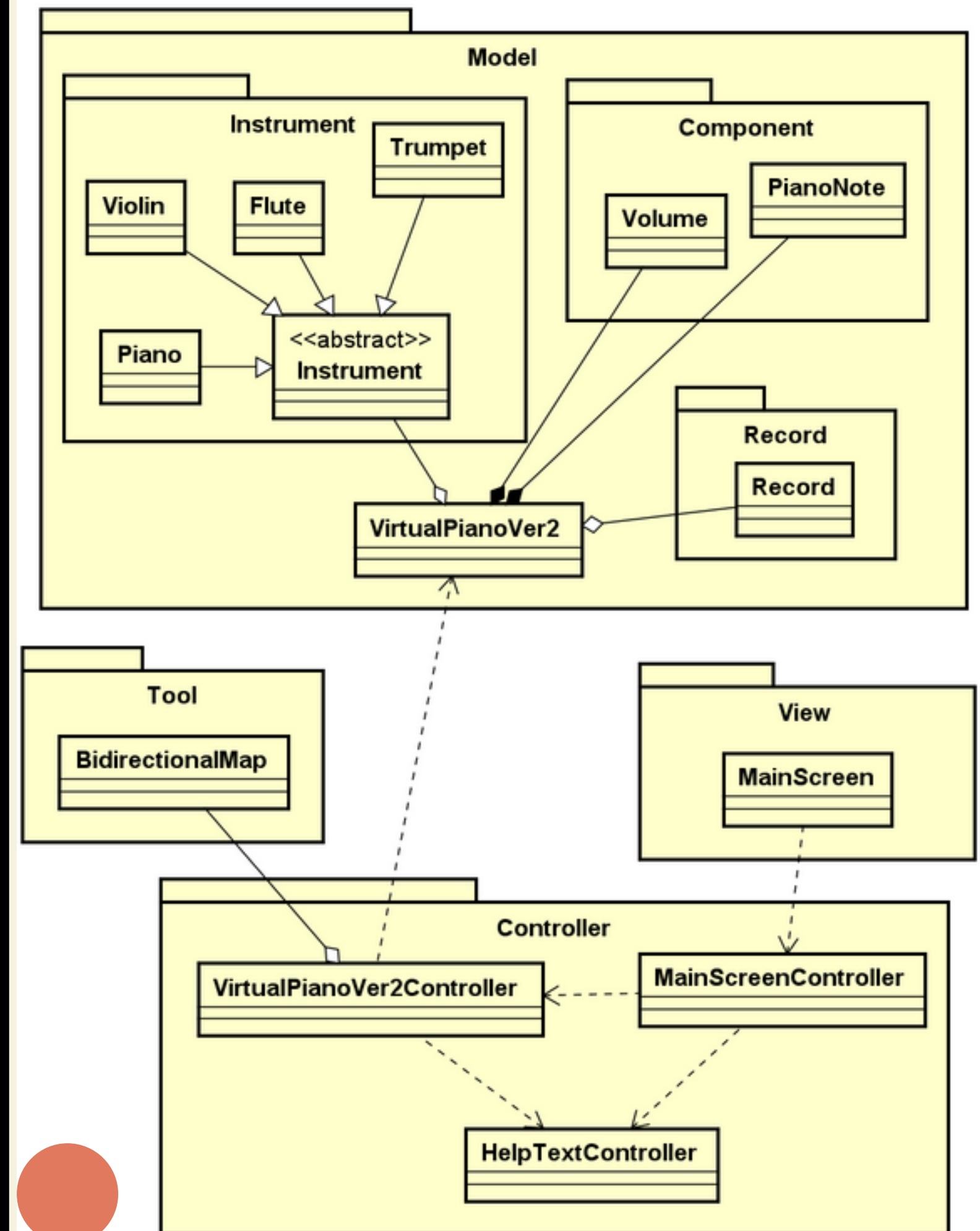User can start to play by clicking, pressing or editing the song themself
- Play a note whenever user touches.
- Notice the user if a key on keyboard cannot play.
- Replay everything we've just played.
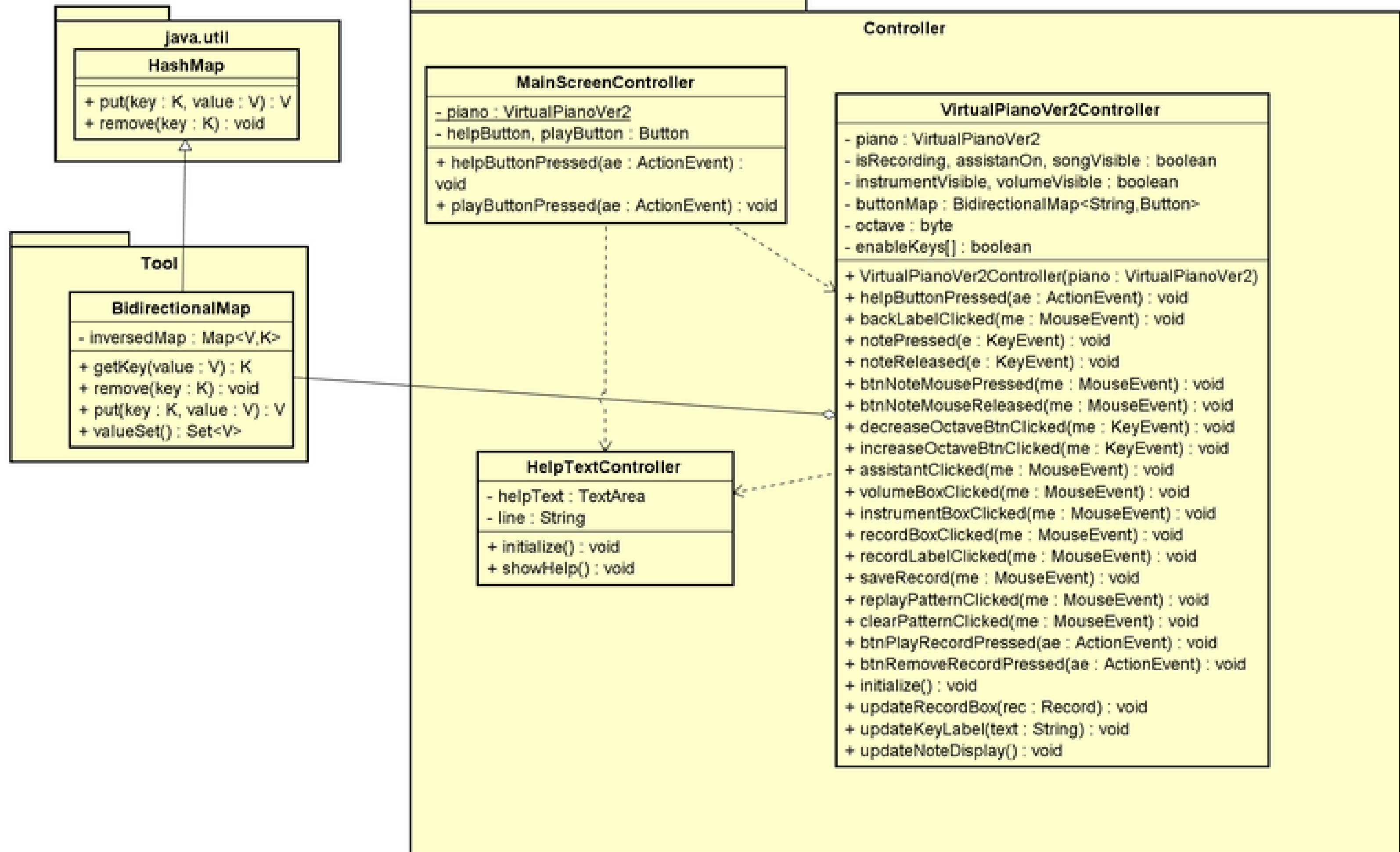- Adjust volume, music style for different experiences.
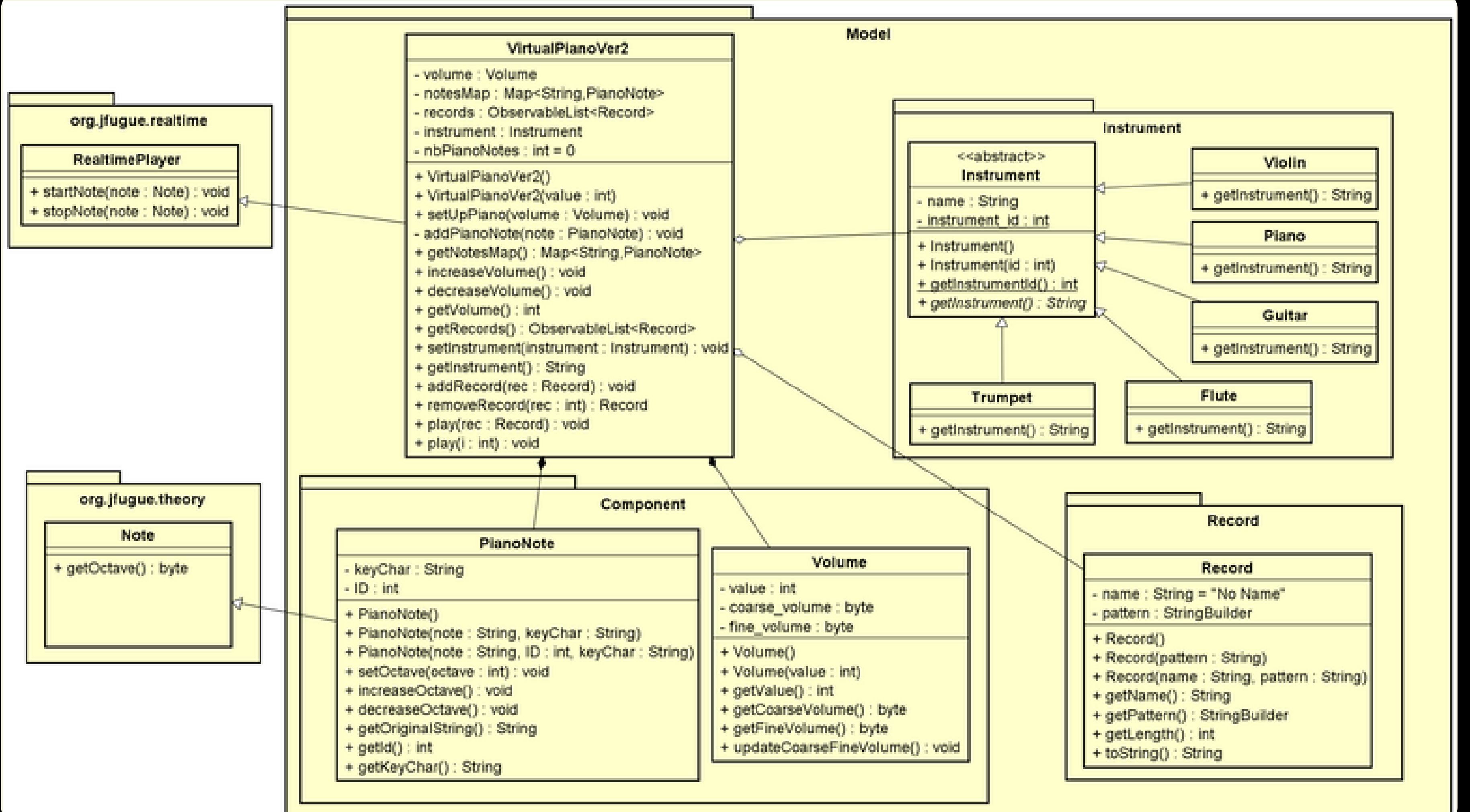
User can view help menu.
User can confirm to exit from the program.

CLASS DIAGRAM

- Package "Model": store a complete piano
- Package "Controllers": store all the screen controllers.
- Package "View": store the "MainScreen" class of the application.
- Package "Tool": store key character and corresponding piano note,

Model

VirtualPianoVer2

- volume : Volume
- notesMap : Map<String,PianoNote>
- records : ObservableList<Record>
- instrument : Instrument
- nbPianoNotes : int = 0

+ VirtualPianoVer2()
+ VirtualPianoVer2(value : int)
+ setUpPiano(volume : Volume) : void
- addPianoNote(note : PianoNote) : void
+ getNotesMap() : Map<String,PianoNote>
+ increaseVolume() : void
+ decreaseVolume() : void
+ getVolume() : int
+ getRecords() : ObservableList<Record>
+ setInstrument(instrument : Instrument) : void
+ getInstrument() : String
+ addRecord(rec : Record) : void
+ removeRecord(rec : int) : Record
+ play(rec : Record) : void
+ play(i : int) : void

org.jfugue.realtime

RealtimePlayer

+ startNote(note : Note) : void
+ stopNote(note : Note) : void

Instrument

<> Instrument

- name : String
- instrument_id : int

+ Instrument()
+ Instrument(id : int)
+ getInstrumentId() : int
+ getInstrument() : String

Violin
+ getInstrument() : String

Piano
+ getInstrument() : String

Guitar
+ getInstrument() : String

Trumpet
+ getInstrument() : String

Flute
+ getInstrument() : String

org.jfugue.theory

Note
+ getOctave() : byte

Component

PianoNote

- keyChar : String
- ID : int

+ PianoNote()
+ PianoNote(note : String, keyChar : String)
+ PianoNote(note : String, ID : int, keyChar : String)
+ setOctave(octave : int) : void
+ increaseOctave() : void
+ decreaseOctave() : void
+ getOriginalString() : String
+ getId() : int
+ getKeyChar() : String

Volume

- value : int
- coarse_volume : byte
- fine_volume : byte

+ Volume()
+ Volume(value : int)
+ getValue() : int
+ getCoarseVolume() : byte
+ getFineVolume() : byte
+ updateCoarseFineVolume() : void

Record

Record

- name : String = "No Name"
- pattern : StringBuilder

+ Record()
+ Record(pattern : String)
+ Record(name : String, pattern : String)
+ getName() : String
+ getPattern() : StringBuilder
+ getLength() : int
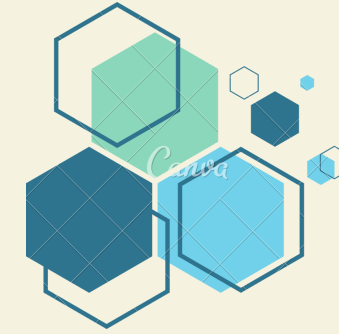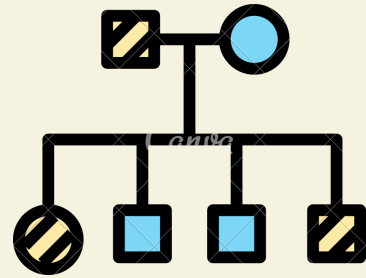+ toString() : String

Encapsulation

- Package Model, class VirtualPianoVer2
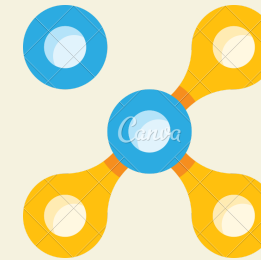- Package Controller, class VirtualPianoVer2Controller use instance of BidirectionalMap

Abstraction

- Package Instrument in package model, 5 class implement abstract method from abstract class Instrument

Inheritance

- Package Model, class VirtualPianoVer2
- Package Controller, class VirtualPianoVer2Controller use instance of BidirectionalMap

Polymorphism

- Package Instrument in package model, 5 class implement abstract method from abstract class Instrument

Demo