Scenarios for HRIS

Use Case #8: User_Views_Stafflist

Requirement:

The user shall be able to view an interactive list of staff employed by the school.

Overview:

When the application first loads, the user shall be able to view an interactive list of staff employed by the school. There will be a tab labeled 'Staff' and this should be available to click upon application startup. The user will select this tab and the interactive list of staff will be shown to the user. It will display the names in the format 'Family Name, Given Name (Title)', as in 'Einstein, Albert (Dr)', ordered alphabetically by family name and then given name. This list should be visually compact and should be able to filter the listed staff based on their category. The user should be able to list all staff, academic, technical, administrative, and casual. It would enhance the system's utility if the user could also filter the list by entering part of a staff member's name. The list contents could be restricted to show only those staff whose given name of family name contains the text entered by the user, ignoring case.

Preconditions:

1. The School Database exists and is populated with data.

Scenario:

Action	Software Reaction
1. User starts the application.	1. The MainView is created with StaffListView and a tab labeled 'Staff'. 2. The system retrieves basic staff details from the database: name in the format of 'Family Name, Given Name (Title)'. 3. Staff names are sorted alphabetically by Family Name and then Given Name, ready to be shown in StaffListView.
2. User selects the tab labeled 'Staff'.	1. The StaffListView is showed to the user in a compact form.
3. User modifies the text in the filter-by-name text field.	1. No reaction; not supporting automatic refresh of list contents.
4. User presses enter key inside the filter-by-name text field.	1. The system updates the current set of filters to include given text or, if search text is blank, removes any existing text from the set of filters. 2. The system creates a temporary list of staff based on the current set of filters (name). if a user has entered name then it will ignore case. 3. The system refreshes the StaffListView with the filtered list.

5. User changes options in the 'category' dropdown and selects an option.	1. The system refreshes the StaffListView with the filter (category).
6. The user selects a staff name on the list.	1. See UC16_User_selects_StaffDetails.

Scenario Notes:

Action 1 is the precondition of all actions. Action 2 needs to be selected for Action 3 to be available. Action 4 depends on Action 3. Action 6 is independent of Action 3 but dependent on Action 1.

Post Conditions

StaffListView displays all or a subset (matching category or name) of staff names inside the first pane of MainView.

Required Views: (GUIs)

MainView
StaffListView

Exceptions:

1. If the database is inaccessible or another unexpected error occurs then the system should respond with a message to this effect, requesting that the user close and restart the software. (In a later release the system should be made robust to such exceptions and be able to reload the database without being restarted.)

Use Cases utilised

UC16_User_selects_StaffDetails

Use Case #16: User_selects_StaffDetails

Requirement:

When the user selects a name in the list shown by StaffListView, the system will show more details about the staff member (referred to as the Staff Details view), which should include: Name; Campus; Phone Number; Room Location; Email Address; Photo; Consultation hours; Table of units he or she is involved with in the current semester.

Overview:

When the user selects a name in the list the system will show more details about the staff member (referred to as the Staff Details view), which should include: Name; Campus; Phone Number; Room Location; Email Address; Photo; Consultation hours; Table of units he or she is involved within the current semester. It would also be useful for each staff member's current availability to be displayed: 'teaching' (with details of the unit code and room) if they are in a timetabled class; 'consulting' if it is during their consultation times; 'free' otherwise. If the user selects a unit code in the table of units then the software could take the user to the timetable view.

Preconditions:

1. School Database exists and is populated with data.
2. StaffListView displays all or a subset (matching category) of staff names (visible).
3. UnitTimetableView or the specifications in UC24 has been already built.

Scenario:

Action	Software Reaction
1. User selects a name in the list shown in the StaffListView.	1. The system retrieves from the database additional information about the staff member including name, campus, phone number, room location, email address, photo, consultation hours, and table of units he or she is involved within the current semester. The current availability details should also be retrieved. 'teaching' (with details of the unit code and room) if they are in a timetabled class; 'consulting' if it is during their consultation times; 'free' otherwise. 2. The system will display the retrieved information in the StaffDetailsView.
2. The user selects a unit code in the table of units.	1. The system will take the user to the UnitTimetableView. (see UC24_User_selects_Unit).

Scenario Notes:

Action 1 must be finished before Action 2 occurs.

Post Conditions

1. The StaffDetailsView shows all the relevant details about the selected Staff name.
2. The user can view the timetable in UnitTimetableView after clicking the unit code in StaffDetailsView.

Required Views: (GUIs)

MainView
StaffDetailsView
UnitTimetableView

Exceptions:

1. If the UnitTimetableView feature is not available, system will show the user an error message stating that.
2. If the database is inaccessible or another unexpected error occurs then the system should respond with a message to this effect, requesting that the user close and restart the software. (In a later release the system should be made robust to such exceptions and be able to reload the database without being restarted.)

Use Cases utilised

UC8_User_Views_Stafflist (precondition)
UC24_User_selects_Unit

Use Case #19: User_shows_ActivityGrid

Requirement:

It would enhance the Staff Details view if the staff member's activity (classes and consultation times) across a week could be displayed in a colour-coded grid.

Overview:

It would enhance the Staff Details view if the staff member's activity (classes and consultation times) across a week could be displayed in a colour-coded grid. This grid should be toggled (displayed or hidden) via a button on the Staff Detail view. The grid should have days of the week (Monday through Friday) as columns and hours of the day (9 am until 4 pm) as rows, with each cell's colour indicating the kind of activity at that time, but no other details are shown. Free time should be shown in white, while teaching and consultation times should be shaded in distinct colours that are distinguishable by those with common forms of colour blindness.

Preconditions:

1. StaffDetailsView is built and visible.
2. The user has clicked a name in the StaffListView and the StaffDetailsView is open.
3. The School Database exists and is populated with data.

Scenario:

Action	Software Reaction
1. The user selects a button named "Show Activity" on the StaffDetailsView.	1. The system will show a colored grid on showing days of the week (Monday through Friday) as columns and hours of the day (9 am until 4 pm) as rows, with each cell's colour indicating the kind of activity at that time, but no other details are shown. Free time should be shown in white, while teaching and consultation times should be shaded in distinct colours that are distinguishable by those with common forms of colour blindness. The button "Show Activity" (the arrow) will have a darker color after being clicked.
2. The user selects the dark-coloured "Show Activity" button.	2. The colored grid on the StaffDetailView will be hidden and the dropdown arrow sign in the "Show Activity" button will have an initial (lighter) colour.

Scenario Notes:

Action 1 needs to happen first, then Action 2 can occur.

Post Conditions

1. The "Show Activity" button on StaffDetailView is available to the user and the user can toggle the colored grid view showing staff member's activity.

Required Views: (GUIs)

MainView
StaffDetailsView

Exceptions:

1. If the database is inaccessible or another unexpected error occurs then the system should respond with a message to this effect, requesting that the user

close and restart the software. (In a later release the system should be made robust to such exceptions and be able to reload the database without being restarted.)

Use Cases utilised

UC16_User_selects_StaffDetails

Use Case #23: User_views_UnitList

Requirement:

The system should be able to generate a list of the units under the control of the School, ordered alphanumerically.

Overview:

There should be an option in MainView where the user will have an option to generate a UnitListView which will generate a list of the units under the control of the school and it will be ordered alphanumerically based on unit code. The list of units shall be formatted as 'UnitCode UnitTitle', as in 'KIT206 Software Design and Development'. The user could also filter the list of units by entering part of a unit code or title. The list contents would be restricted to show only those units whose code or title contains the text entered by the user, ignoring case. Search queries including both code and title do not need to be supported.

Preconditions:

1. The School Database exists and is populated with data.
2. There is an option/button to generate the unit list in the MainView.

Scenario:

Action	Software Reaction
1. User selects the "Generate Unit List" in MainView.	1. The System will load all the unit codes with their title from the database. As 'UnitCode UnitTitle'. 2. The System will show the unit list in the UnitListView and shows a text box "filter-by-unit" for searching as well.
2. The user types unit code or name in the "filter-by-unit" text box.	1. Nothing happens. Desired Data will not be updated automatically.
3. The user presses enter in the "filter-by-unit" text box.	1. The list contents in the unit list will be then restricted to show only those units whose code or title contains the text entered by the user, ignoring case.

Scenario Notes:

This scenario is being developed independently from staff details options and the user would be able to open it from MainView. Search queries including both code and title do not need to be supported. Action 2 needs to occur before Action 3.

Post Conditions

1. The user views all unit code and their name as a list (ordered alphanumerically) in the UnitListView.

Required Views: (GUIs)

MainView
UnitListView

Exceptions:

1. If the database is inaccessible or another unexpected error occurs then the system should respond with a message to this effect, requesting that the user close and restart the software. (In a later release the system should be made robust to such exceptions and be able to reload the database without being restarted.)

Use Cases utilised

none

Use Case #24: User_selects_Unit

Requirement:

Selecting a unit from the unit list should bring up a list of classes for that unit, ordered chronologically.

Overview:

The user should be able to select a unit from this list which should bring up a list of classes for that unit and this should be ordered chronologically. This view should be tabular, showing the following information about the selected unit in each column: day; start and end time in 24-hour format, such as '12:00–14:00' or '12:00–13:50'; type (Lecture, Tutorial, Practical, Workshop); room location; campus; staff member. The user could go from a timetable entry to the Staff Detail view for the relevant staff member, just as if selecting that person in the Staff List. The user will be able to filter the timetable so that it displays information for a single campus.

Preconditions:

1. UnitListView displays the generated unit list correctly.
2. School database exists and data is properly retrieved from the database.

Scenario:

Action	Software Reaction
1. The user selects a unit name from the unit list on the UnitListView.	1. The system retrieves data from the database about the classes of that unit and orders them chronologically. 2. The system retrieves the data of each of those classes including the date, start and end time in 24-hour format, such as '12:00–14:00' or '12:00–13:50'; type (Lecture, Tutorial, Practical, Workshop); room location; campus; staff member. 3. The System shows all the information in a tabular view named "UnitTimetableView".
2.The user selects a single campus UnitTimetableView by a filter-by-campus dropdown menu.	1. The "UnitTimetableView" only show the subset of classes which are of the campus entered by the user in the filter-by-campus dropdown menu

3. The user selects any timetable entry in the UnitTimetableView.	1. The system will open the StaffDetailView of that relevant staff (who is taking classes during the selected time)

Scenario Notes:

Action 1 needs to happen before Action 2 and Action 3.

Post Conditions

1. The user is able to select a unit name from the unit list on the UnitListView.
2. The UnitTimetableView will appear after selecting a unit.
3. The StaffDetailView will appear after clicking any time entry on UnitTimetableView.
4. Filter option is available to the user for filtering classes by single campus.

Required Views: (GUIs)

UnitListView
UnitTimetableView
StaffDetailView

Exceptions:

1. Will show error message if data became corrupt or could not be retrieved from the database.

Use Cases utilised

UC23_User_views_UnitList
UC16_User_selects_StaffDetails

Use Case #28: User_generates_HeatMap

Requirement:

The application will generate 'heat maps' of activity across the week.

Overview:

There will be a menu button or tab from where all the heat maps will be accessed. There will be options to create heatmap only in Hobart or only in Launceston or heatmap for all the ICT Discipline classes across both campuses. There will be two types of heatmaps that could be generated- one only for unit classes and the other for consultation times. There will be a button beside the menu button which will allow the user to change the primary color of the heatmap.

Preconditions:

1. The school database exists and populated with proper data for the heatmap to be generated.

Scenario:

Action	Software Reaction
1. The user selects the menu button named "HeatMap".	1. The System shows the user two options in a dropdown menu- "Unit Classes" and "Consultation times".
2. The user chooses an option between "Unit Classes" and "Consultation times".	1. The system shows another dropdown menu to the user that has three options for selecting campus- to generate the heatmap for Hobart, to

	generate the heatmap for Launceston and to generate the heatmap for all campus in ICT Discipline.
3. The user selects or chooses an option from the campus options.	1. The HeatMap will be generated and be displayed on the "HeatMapView". It will be displayed as a grid of coloured cells, with columns for each day of the workweek (Monday through Friday) and rows for the starting hours 9 am to 4 pm. Each cell in which zero activities occur should be blank, while cells representing one or more activities should contain the number of co-occurring activities. The shading of each cell should indicate the number of events occurring at that time, with white indicating no events and a solid colour indicating the greatest number of events within the set of events being displayed. Intermediate values should be assigned a colour intermediate between white and the selected solid colour.
4. The user selects the button named "Change primary colour" beside the "HeatMap" menu button.	1. The system will show a popup below the change primary colour menu where different colour palettes will be displayed.
5. The user selects a colour from the colour palettes.	1. The system will store the user-selected color and make it the primary color when the heatmap will be generated next time.

Scenario Notes:

Action 1, Action 2, and Action 3 needs to be in order. Action 4 does not need to be in order with the previous set of Action but Action 4 needs to happen before Action 5.

Post Conditions

1. The desired heat map is displayed on the HeatMapView for the user.

Required Views: (GUIs)

MainView
HeatMapView

Exceptions:

1. If any error occurs while retrieving data from the database, the user should be notified.
2. If the primary colour is not set properly, the user should see an error message or notice.

Use Cases utilised

none

Use Case #37: User_generates_ClashMap

Requirement:

The application could generate a variant of the heat map that shows clashes between a unit's classes and the consultation time of staff involved in teaching that unit.

Overview:

There will be a similar type of utility as a heatmap which will be called a "Clash Map". This option will be available to the user through a button on UnitTimetableView. This will use data for the currently displayed classes and staff who teach them (that is both campuses, Hobart only or Launceston only). The cells in the grid should use the following colour scheme: white if no class or staff consultation occurs on that hour and day; bright green if that time contains either a class or consultation, but not both; and red if it contains both consultation and a class. Cells representing clashes should also include the text 'clash'.

Preconditions:

1. UnitTimetableView is displayed properly to the user (This means the user had already selected a unit).

Scenario:

Action	Software Reaction
1. The user selects the "Generate Clash Map" button in the UnitTimetableView.	1. The system uses the data which is displayed on the UnitTimetableView currently and produces clashmap using the data. 2. The system will then display the grids in the "ClashMapView" (fourth pane). The cells in the grid should use the following colour scheme: white if no class or staff consultation occurs on that hour and day; bright green if that time contains either a class or consultation, but not both; and red if it contains both consultation and a class. Cells representing clashes should also include the text 'clash'.

Scenario Notes:

This scenario will use already entered data by the user in the UnitTimetableView including data entered in the filter text box (such as campus information).

Post Conditions

1. ClashMapView will display the clashmap to the user.

Required Views: (GUIs)

MainView
UnitTimetableView
ClashMapView

Exceptions:

1. If the ClashMapView fails to generate clashmap or crashes, it should let the user know.

Use Cases utilised

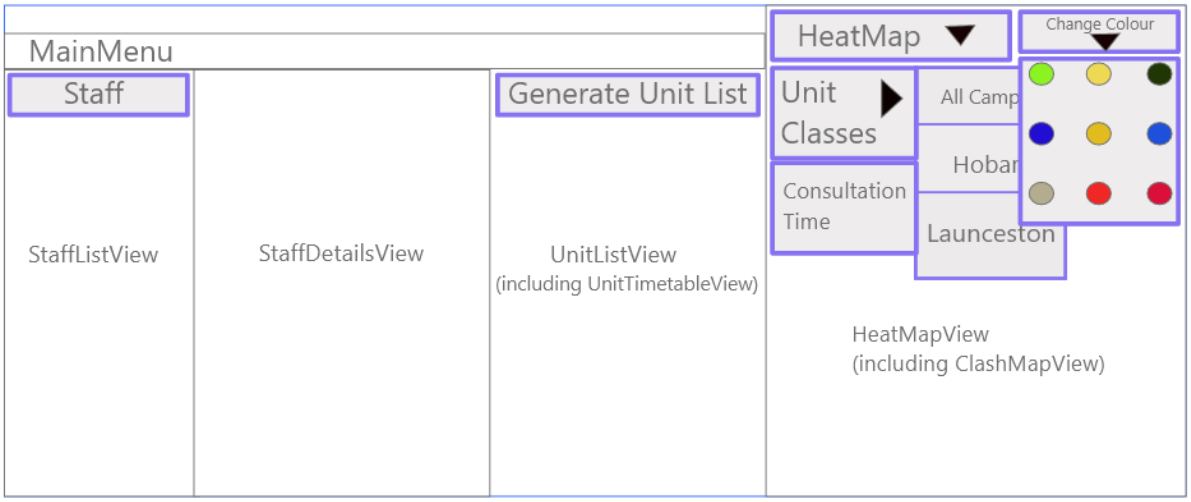
UC24_User_selects_Unit

A brief description about the low fidelity prototype

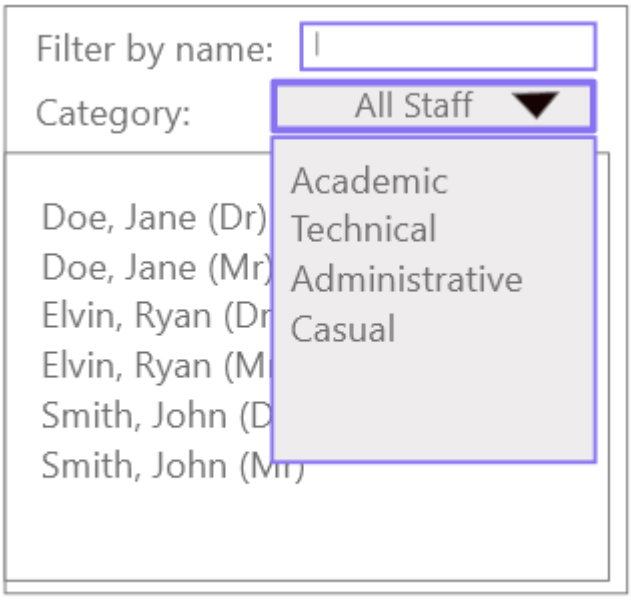
The user will be able to see the MainView when they start the HRIS application. In this main view, there are four panes including a separate pane for map generation. On the very left side of the MainView there is the StaffListView (first pane) with a tab named Staff. User can click this tab and see the Staff names in the StaffListView and the user will be able to filter them.

Then when the user clicks a Staff name inside the StaffListView, they will be able to see the StaffDetails in the next pane which is allocated for the StaffDetailsView. The third pane is for the UnitListView and the UnitTimetableView. No matter from where the user chooses the option to view a unit time table (either after clicking 'Generate Unit List' and clicking a unit from the list or after selecting a unit code from StaffDetailsView), they will see the timetable in the third pane. If they click the 'Generate Unit List' tab at the top of the pane, they will be able to see unit lists in the third pane and be able to filter them.

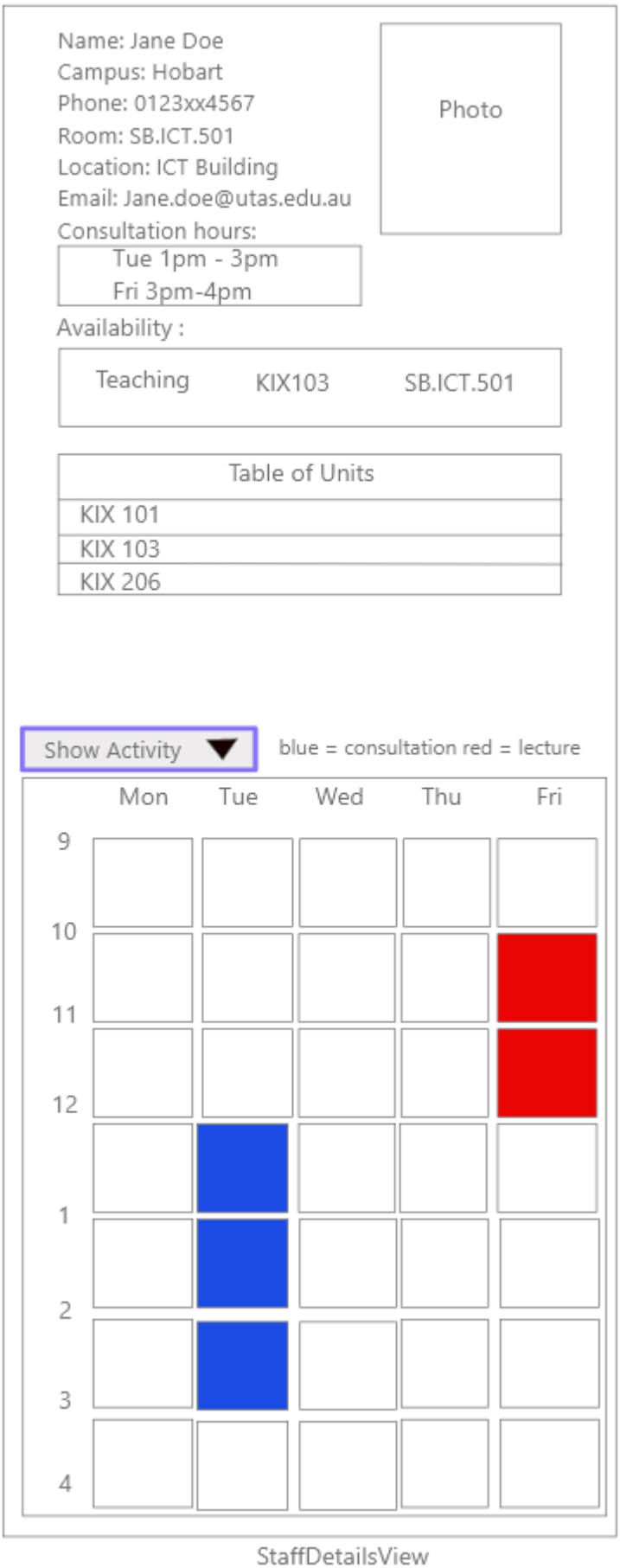
The last pane in the MainView is for both the HeatMap and the ClashMap. The user can generate HeatMap and select color or other options as shown in the prototype. But if the user decides to generate a ClashMap from UnitTimetableView, they will still generate a ClashMap in the fourth pane. The view can be changed again by clicking HeatMap button.



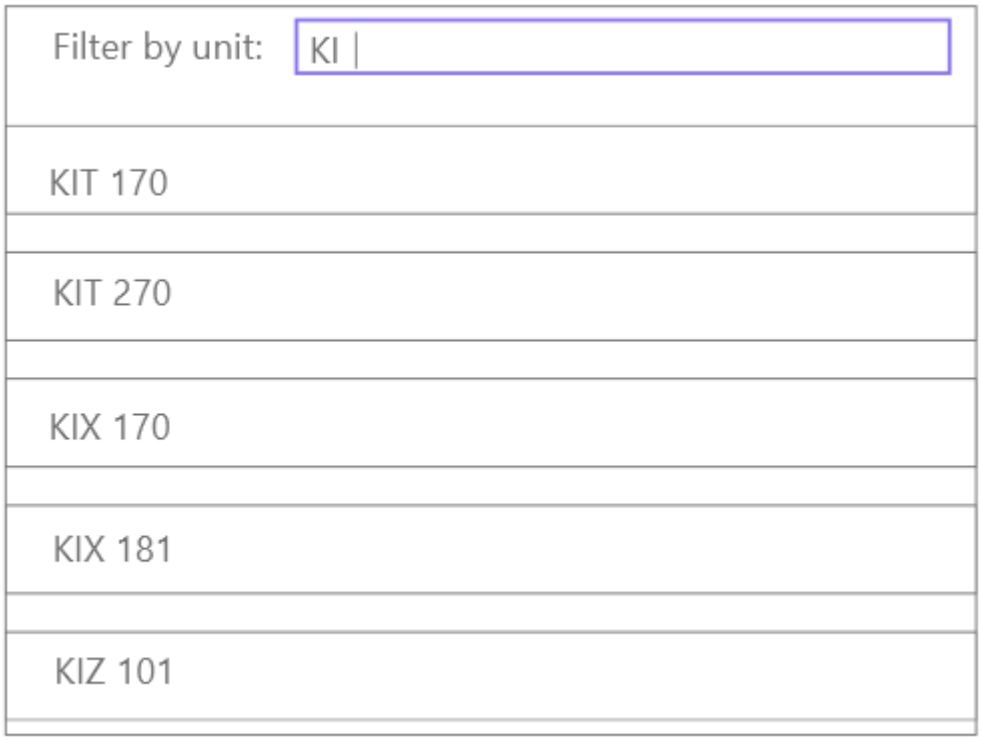
MainView



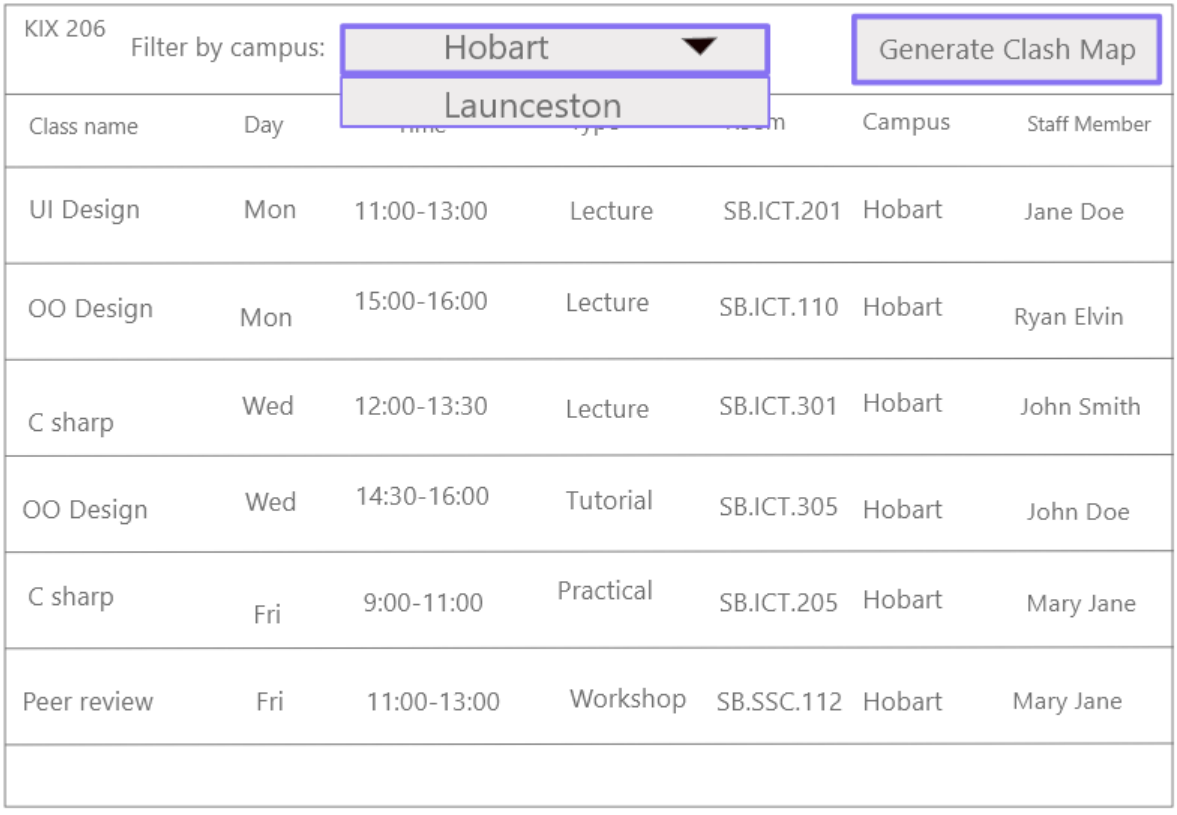
StaffListView



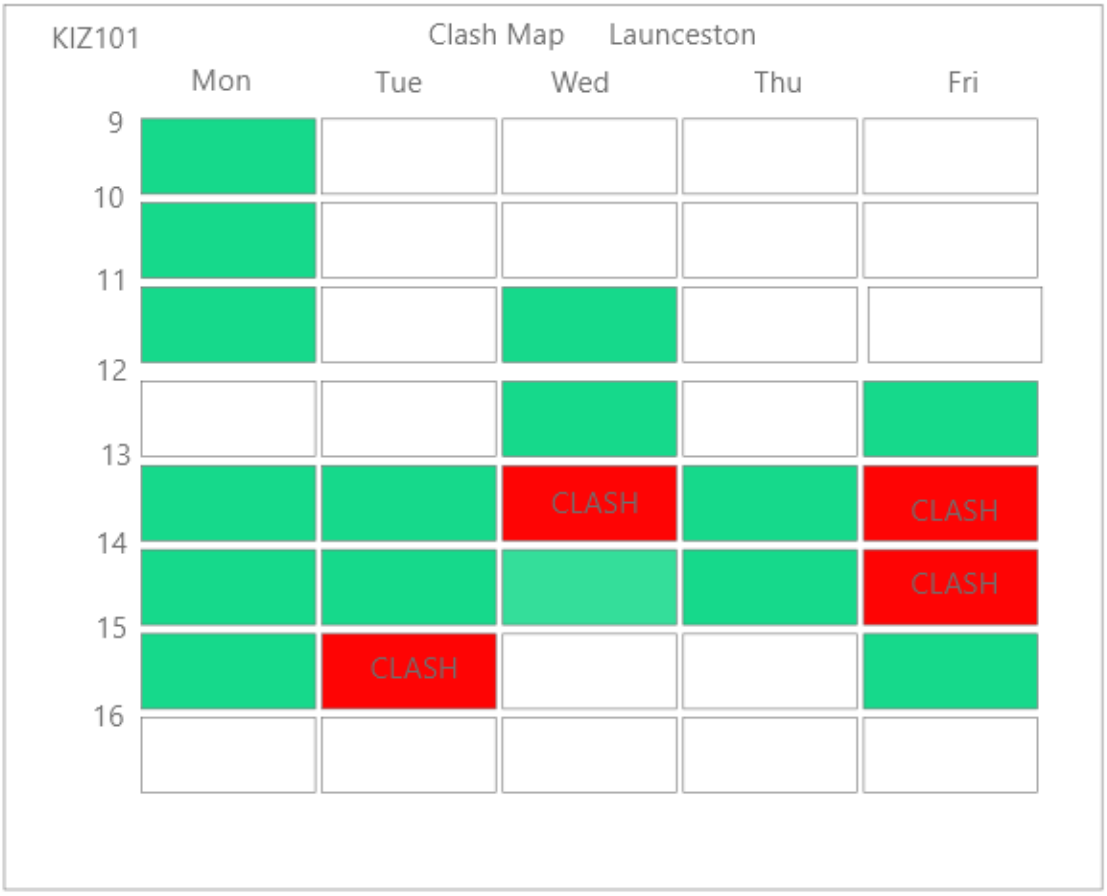
StaffDetailsView



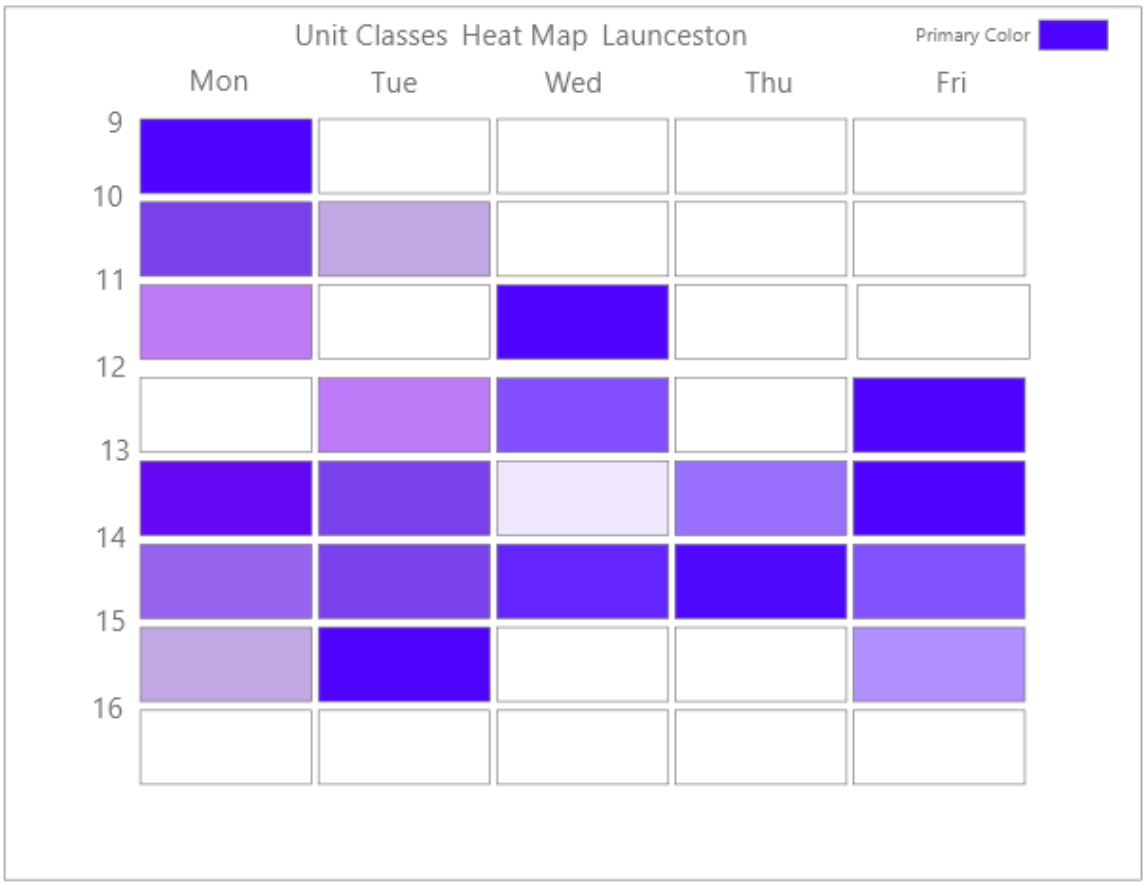
UnitListView



UnitTimetableView

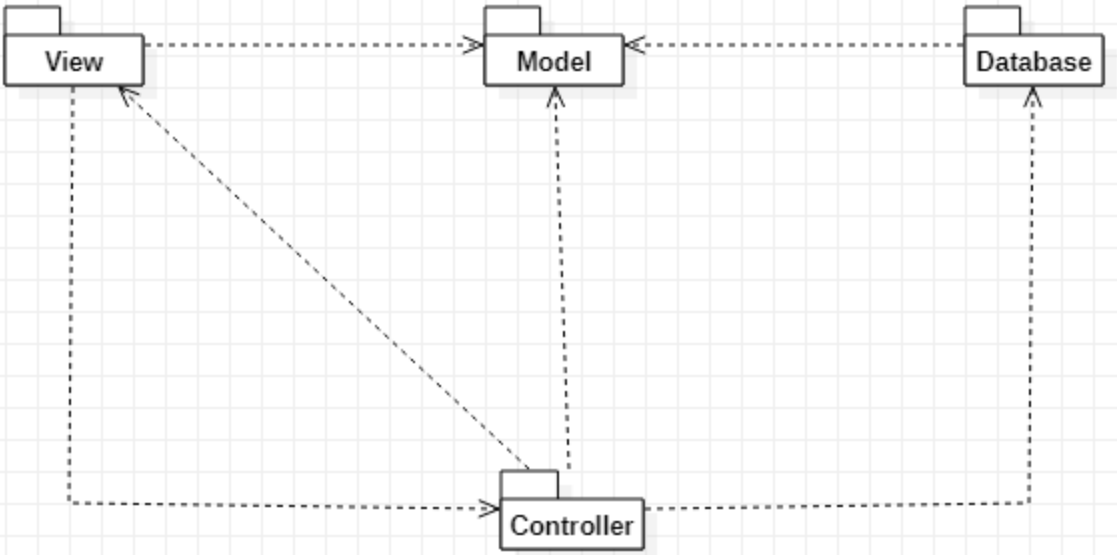


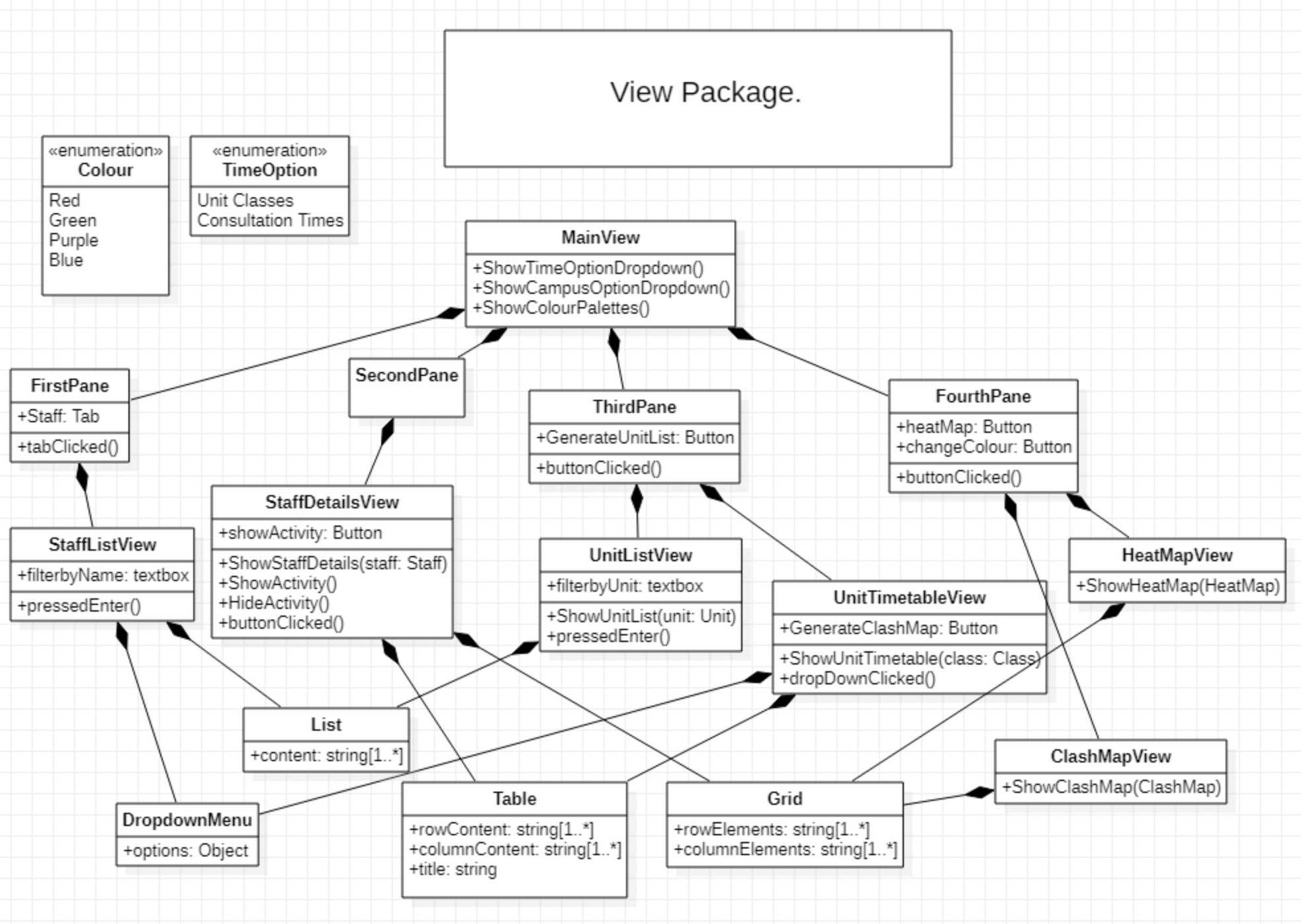
ClashMapView

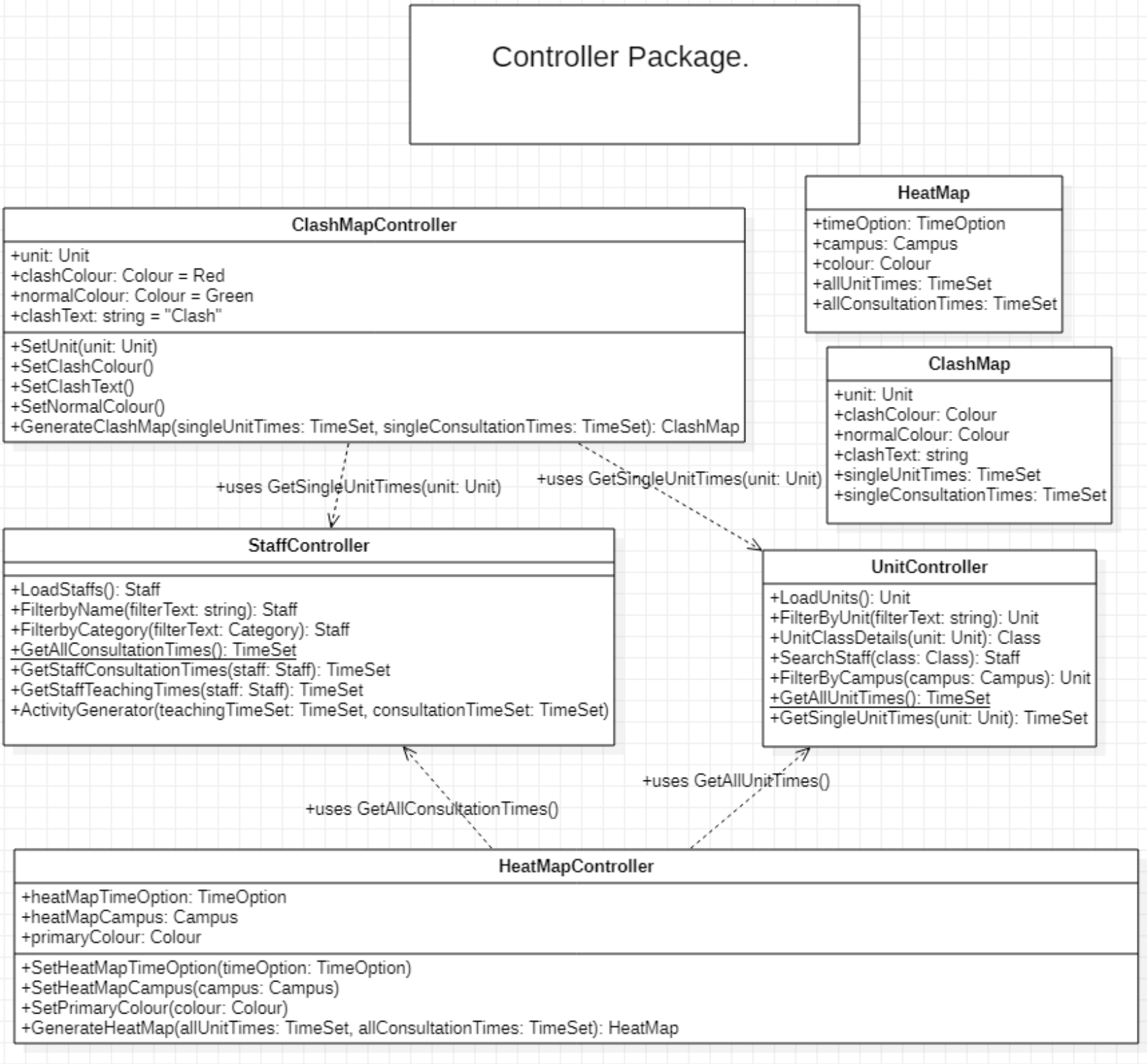


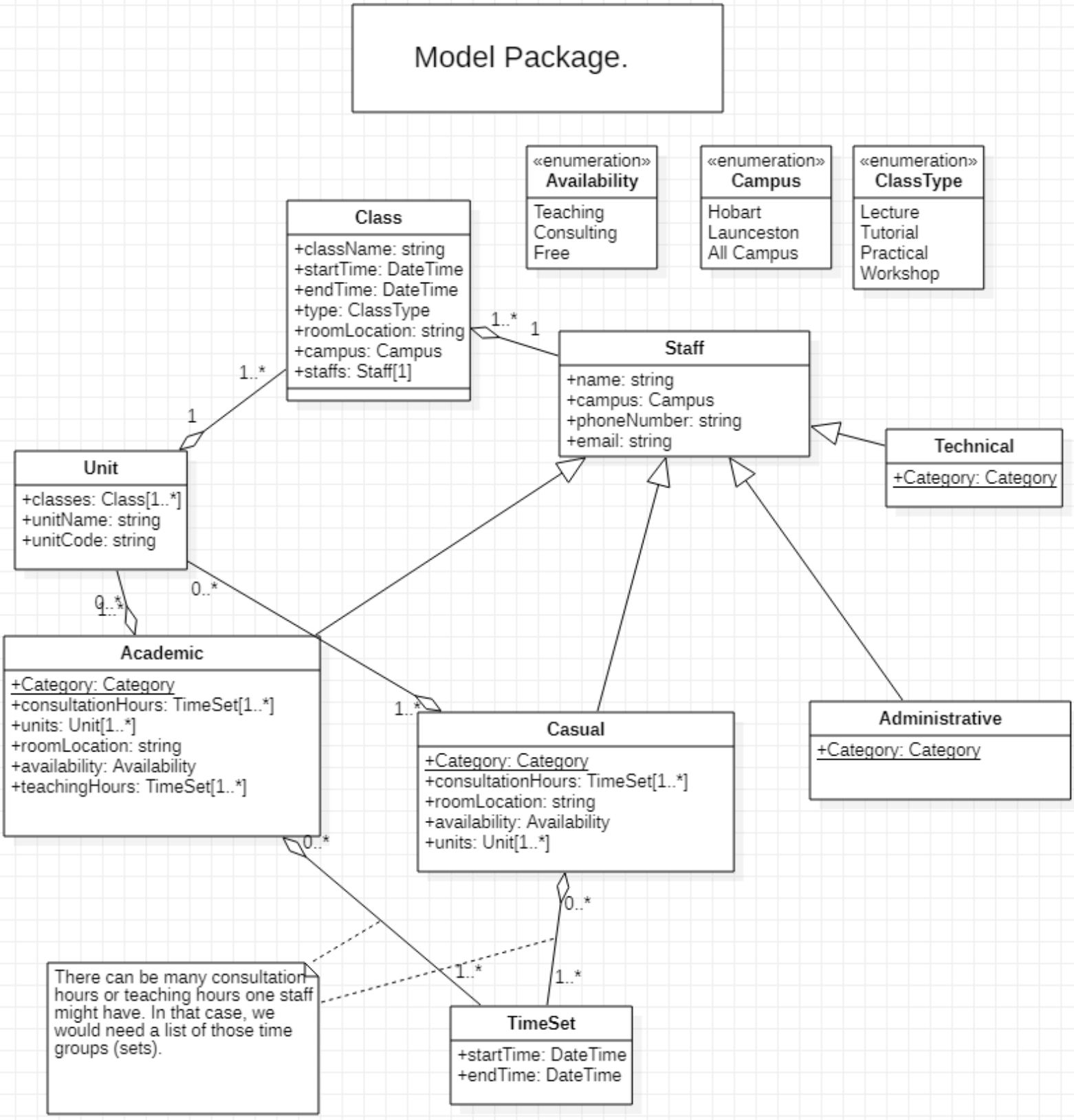
HeatMapView

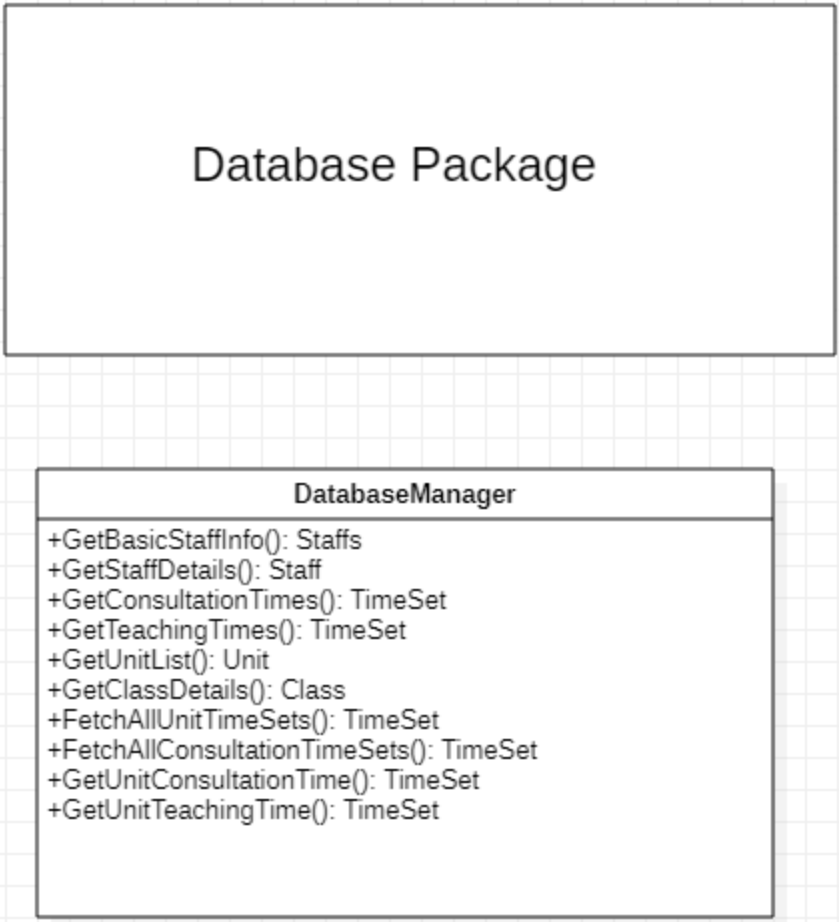
Package Diagram

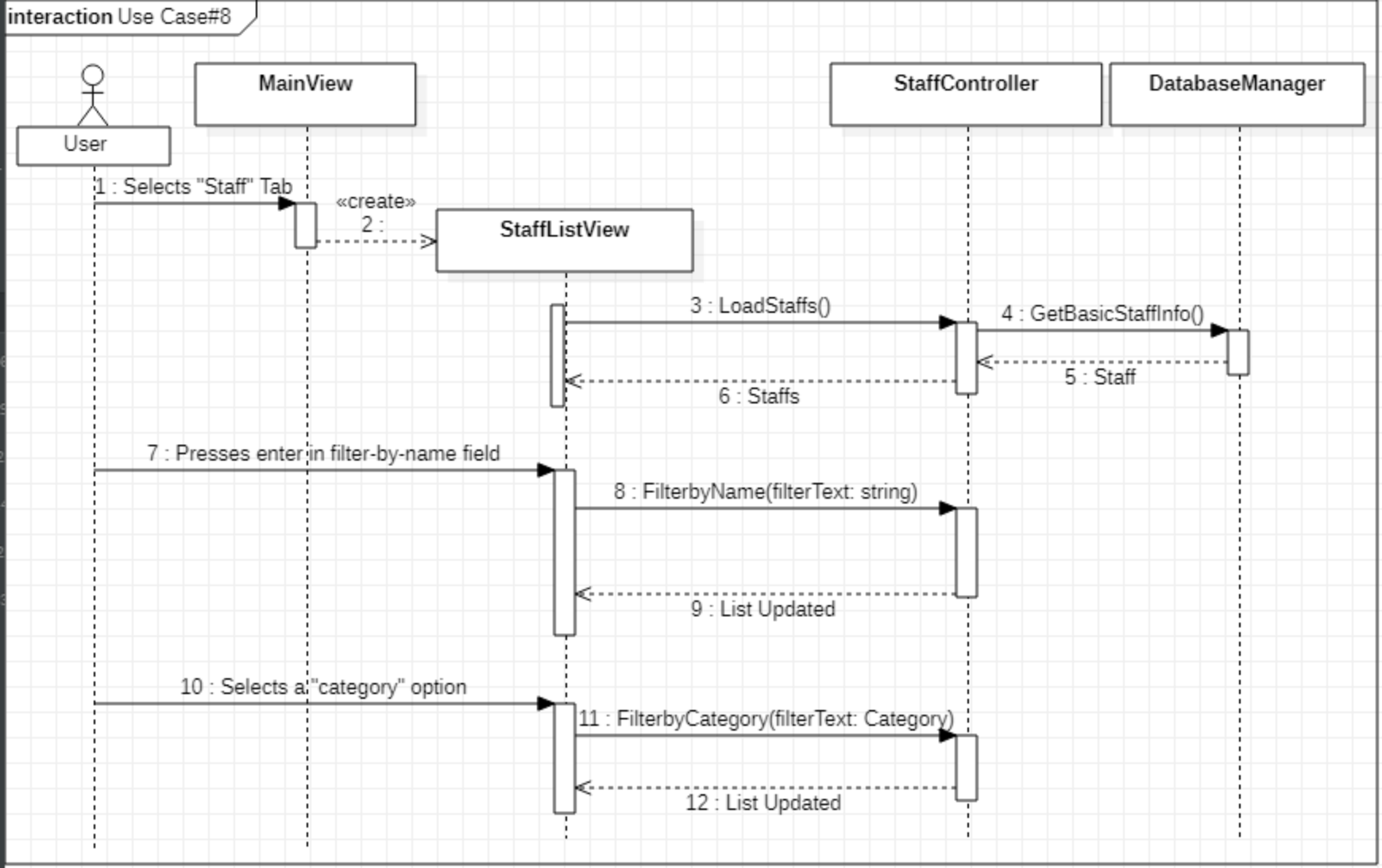


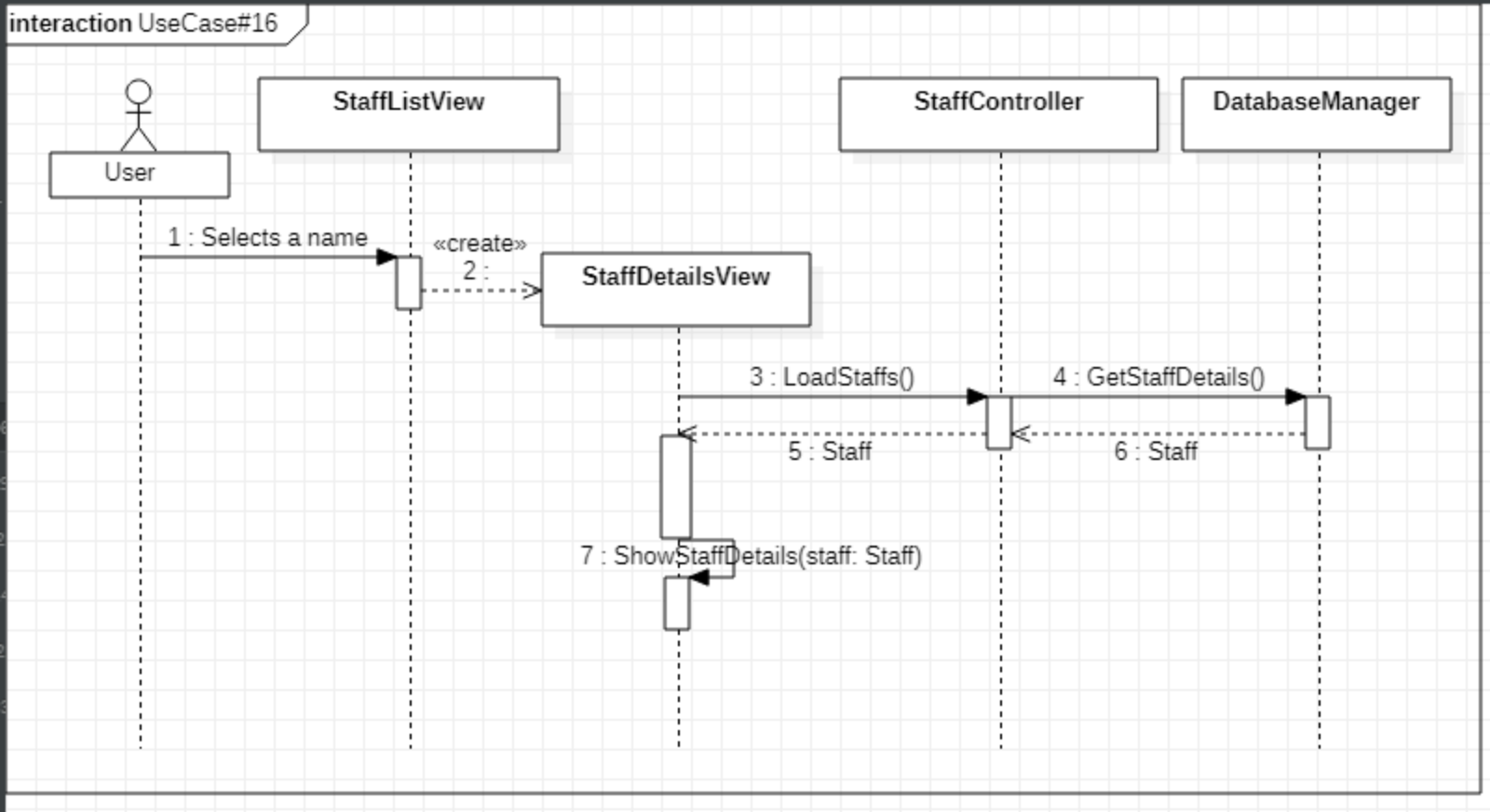

View Package.
«enumeration»
Colour
Red
Green
Purple
Blue
«enumeration»
TimeOption
Unit Classes
Consultation Times
MainView
+ShowTimeOptionDropdown()
+ShowCampusOptionDropdown()
+ShowColourPalettes()
FirstPane
+Staff: Tab
+tabClicked()
SecondPane
ThirdPane
+GenerateUnitList: Button
+buttonClicked()
FourthPane
+heatMap: Button
+changeColour: Button
+buttonClicked()
StaffDetailsView
+showActivity: Button
+ShowStaffDetails(staff: Staff)
+ShowActivity()
+HideActivity()
+buttonClicked()
StaffListView
+filterbyName: textbox
+pressedEnter()
UnitListView
+filterbyUnit: textbox
+ShowUnitList(unit: Unit)
+pressedEnter()
HeatMapView
+ShowHeatMap(HeatMap)
UnitTimetableView
+GenerateClashMap: Button
+ShowUnitTimetable(class: Class)
+dropDownClicked()
List
+content: string[1..*]
ClashMapView
+ShowClashMap(ClashMap)
DropdownMenu
+options: Object
Table
+rowContent: string[1..*]
+columnContent: string[1..*]
+title: string
Grid
+rowElements: string[1..*]
+columnElements: string[1..*]

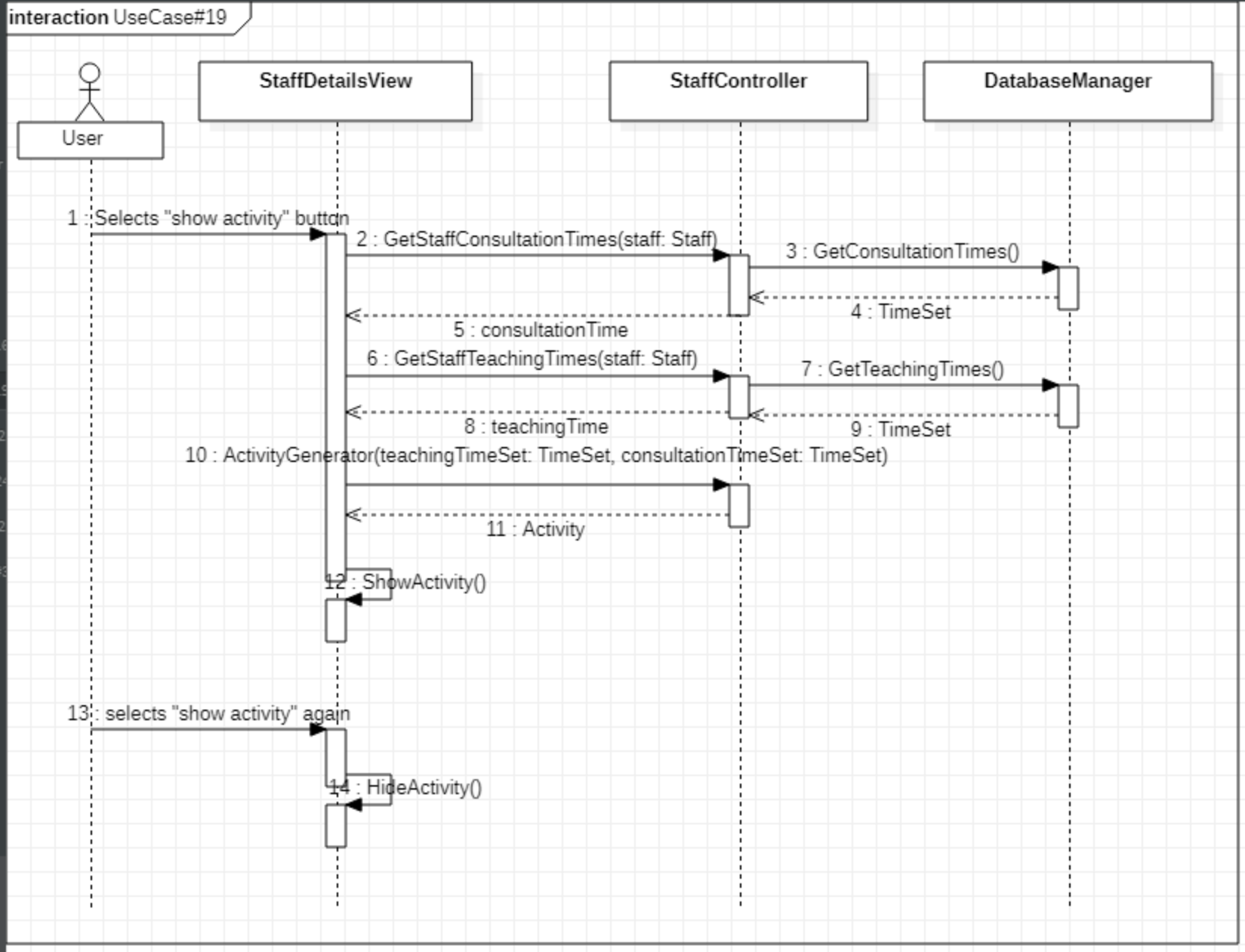

Controller Package.
HeatMap
+timeOption: TimeOption
+campus: Campus
+colour: Colour
+allUnitTimes: TimeSet
+allConsultationTimes: TimeSet
ClashMapController
+unit: Unit
+clashColour: Colour = Red
+normalColour: Colour = Green
+clashText: string = "Clash"
+SetUnit(unit: Unit)
+SetClashColour()
+SetClashText()
+SetNormalColour()
+GenerateClashMap(singleUnitTimes: TimeSet, singleConsultationTimes: TimeSet): ClashMap
ClashMap
+unit: Unit
+clashColour: Colour
+normalColour: Colour
+clashText: string
+singleUnitTimes: TimeSet
+singleConsultationTimes: TimeSet
+uses GetSingleUnitTimes(unit: Unit)
+uses GetSingleUnitTimes(unit: Unit)
StaffController
+LoadStaffs(): Staff
+FilterbyName(filterText: string): Staff
+FilterbyCategory(filterText: Category): Staff
+GetAllConsultationTimes(): TimeSet
+GetStaffConsultationTimes(staff: Staff): TimeSet
+GetStaffTeachingTimes(staff: Staff): TimeSet
+ActivityGenerator(teachingTimeSet: TimeSet, consultationTimeSet: TimeSet)
UnitController
+LoadUnits(): Unit
+FilterByUnit(filterText: string): Unit
+UnitClassDetails(unit: Unit): Class
+SearchStaff(class: Class): Staff
+FilterByCampus(campus: Campus): Unit
+GetAllUnitTimes(): TimeSet
+GetSingleUnitTimes(unit: Unit): TimeSet
+uses GetAllUnitTimes()
+uses GetAllConsultationTimes()
HeatMapController
+heatMapTimeOption: TimeOption
+heatMapCampus: Campus
+primaryColour: Colour
+SetHeatMapTimeOption(timeOption: TimeOption)
+SetHeatMapCampus(campus: Campus)
+SetPrimaryColour(colour: Colour)
+GenerateHeatMap(allUnitTimes: TimeSet, allConsultationTimes: TimeSet): HeatMap

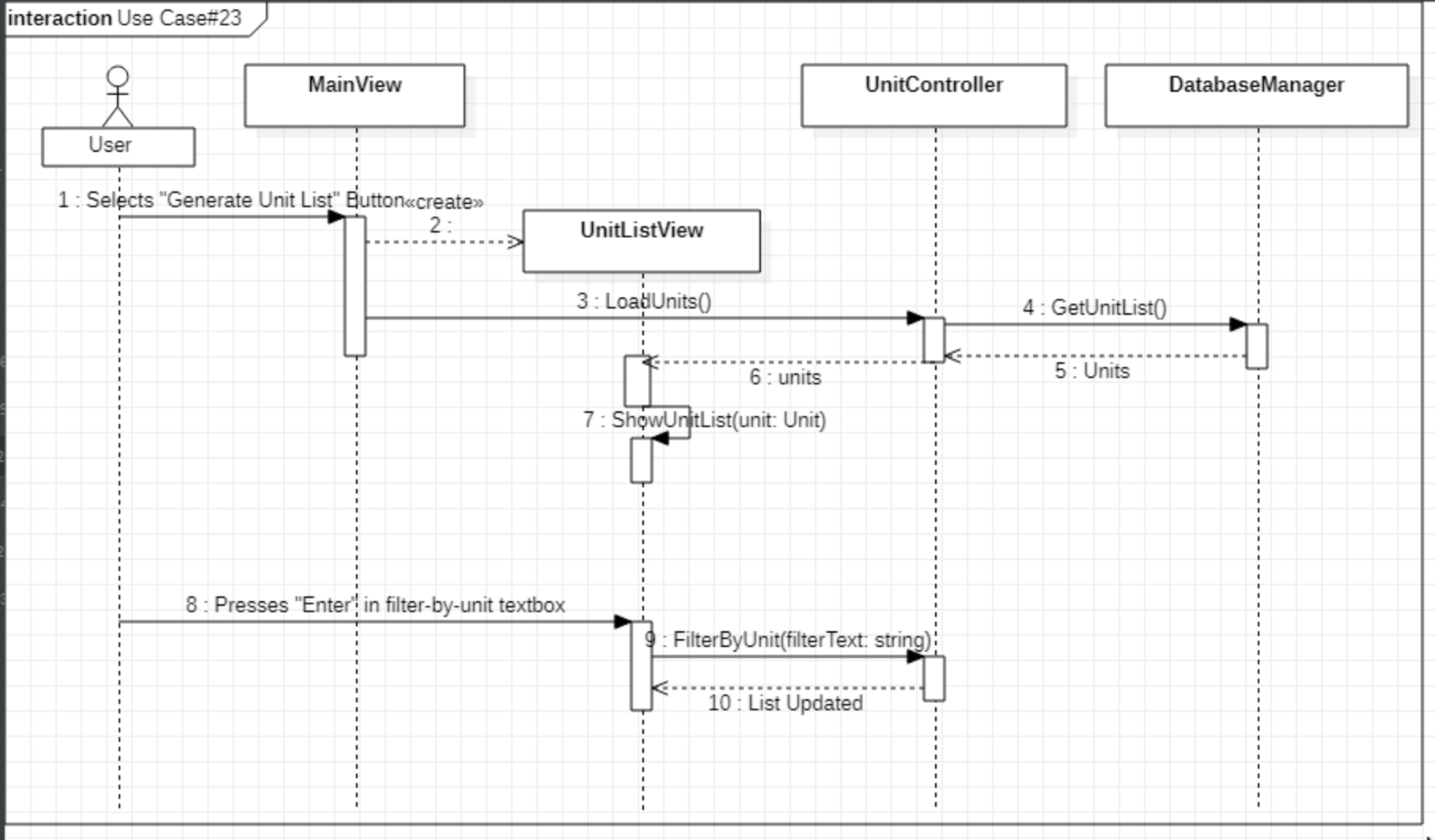

Model Package.
«enumeration»
Availability
Teaching
Consulting
Free
«enumeration»
Campus
Hobart
Launceston
All Campus
«enumeration»
ClassType
Lecture
Tutorial
Practical
Workshop
Class
+className: string
+startTime: DateTime
+endTime: DateTime
+type: ClassType
+roomLocation: string
+campus: Campus
+staffs: Staff[1]
Staff
+name: string
+campus: Campus
+phoneNumber: string
+email: string
Technical
+Category: Category
Unit
+classes: Class[1..*]
+unitName: string
+unitCode: string
Academic
+Category: Category
+consultationHours: TimeSet[1..*]
+units: Unit[1..*]
+roomLocation: string
+availability: Availability
+teachingHours: TimeSet[1..*]
Casual
+Category: Category
+consultationHours: TimeSet[1..*]
+roomLocation: string
+availability: Availability
+units: Unit[1..*]
Administrative
+Category: Category
TimeSet
+startTime: DateTime
+endTime: DateTime
There can be many consultation hours or teaching hours one staff might have. In that case, we would need a list of those time groups (sets).
1..*
1
0..*

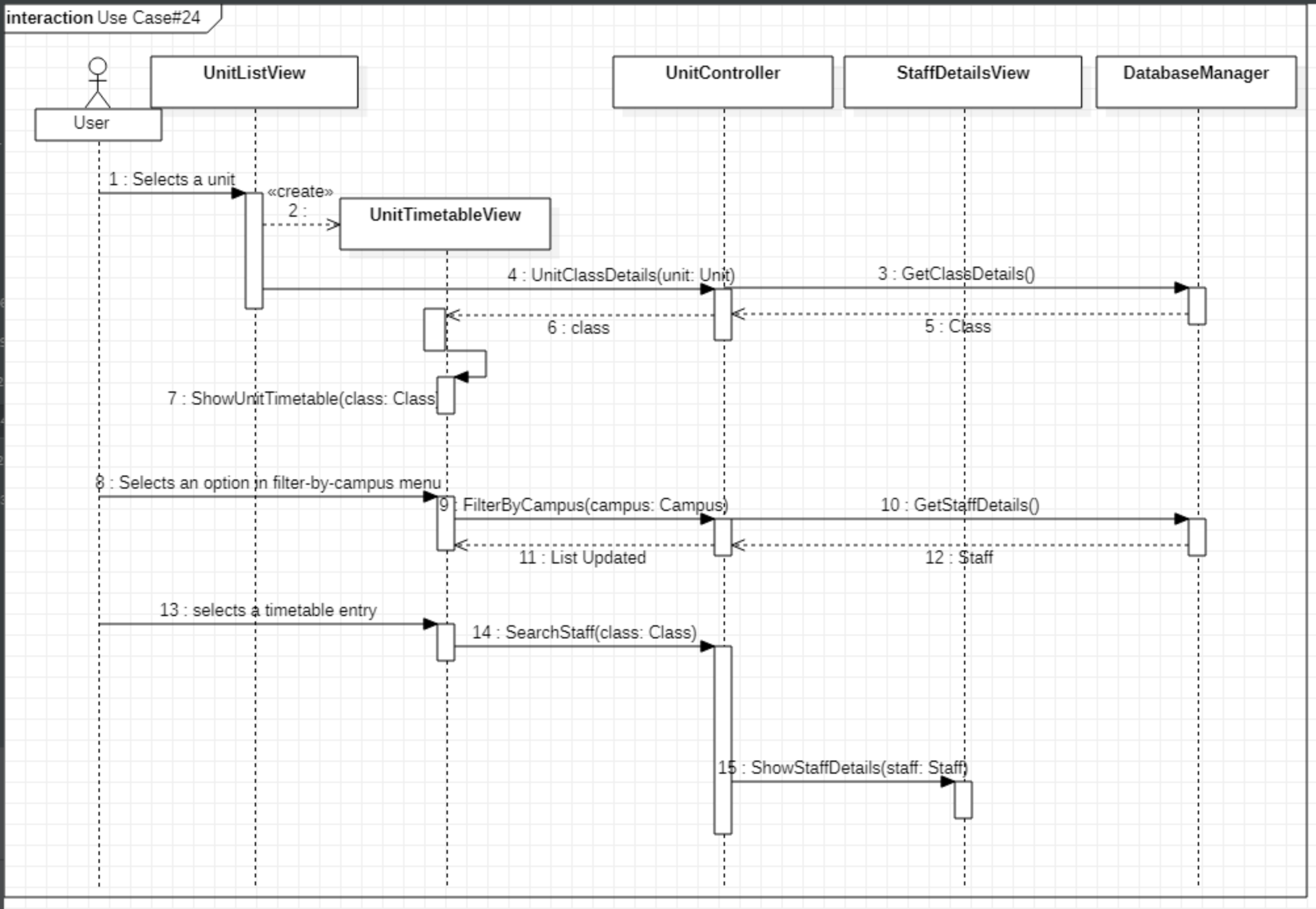

Database Package
DatabaseManager
+GetBasicStaffInfo(): Staffs
+GetStaffDetails(): Staff
+GetConsultationTimes(): TimeSet
+GetTeachingTimes(): TimeSet
+GetUnitList(): Unit
+GetClassDetails(): Class
+FetchAllUnitTimeSets(): TimeSet
+FetchAllConsultationTimeSets(): TimeSet
+GetUnitConsultationTime(): TimeSet
+GetUnitTeachingTime(): TimeSet

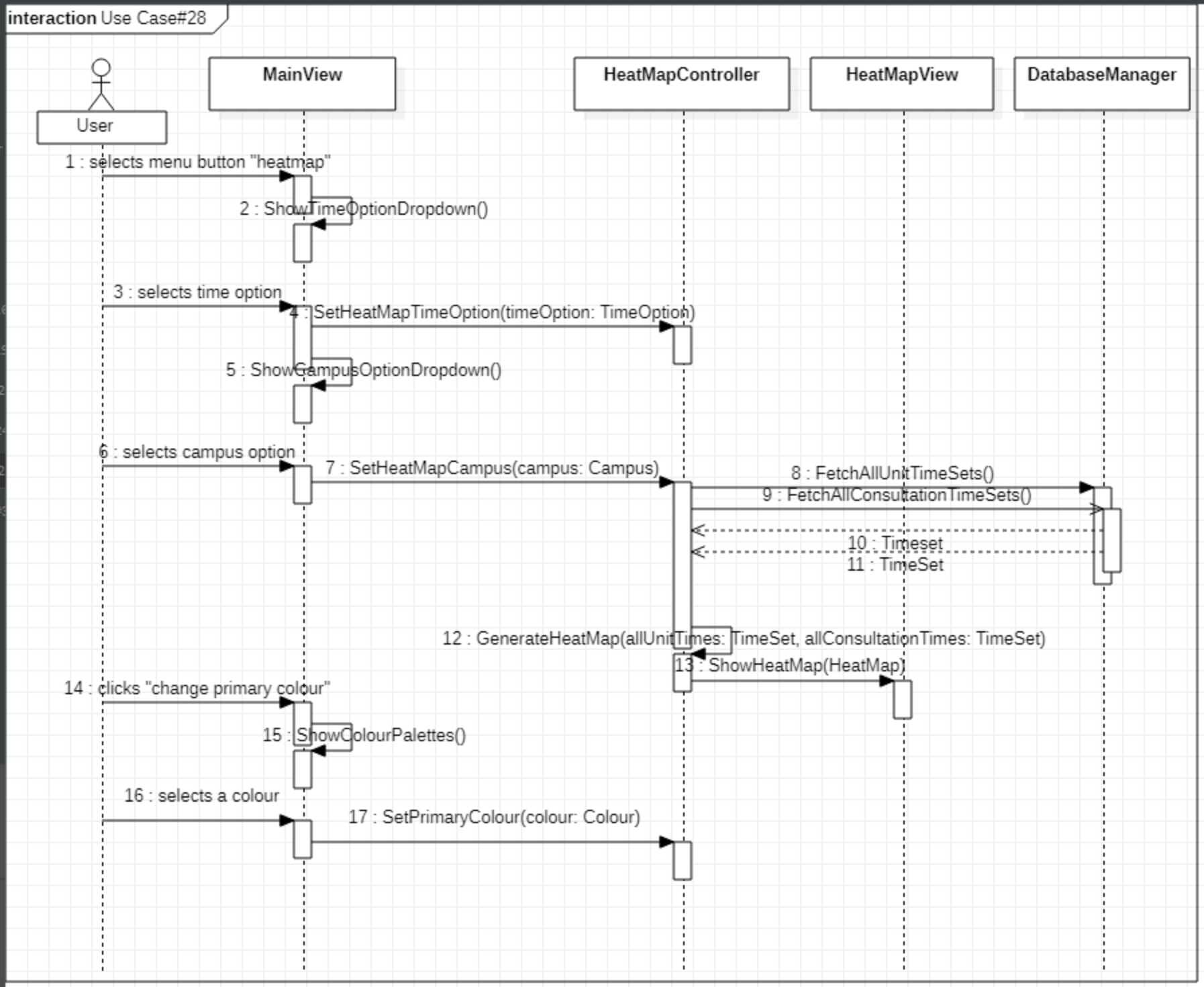

interaction Use Case#8
User
MainView
StaffController
DatabaseManager
1 : Selects "Staff" Tab
«create»
2 :
StaffListView
3 : LoadStaffs()
4 : GetBasicStaffInfo()
5 : Staff
6 : Staffs
7 : Presses enter in filter-by-name field
8 : FilterbyName(filterText: string)
9 : List Updated
10 : Selects a "category" option
11 : FilterbyCategory(filterText: Category)
12 : List Updated

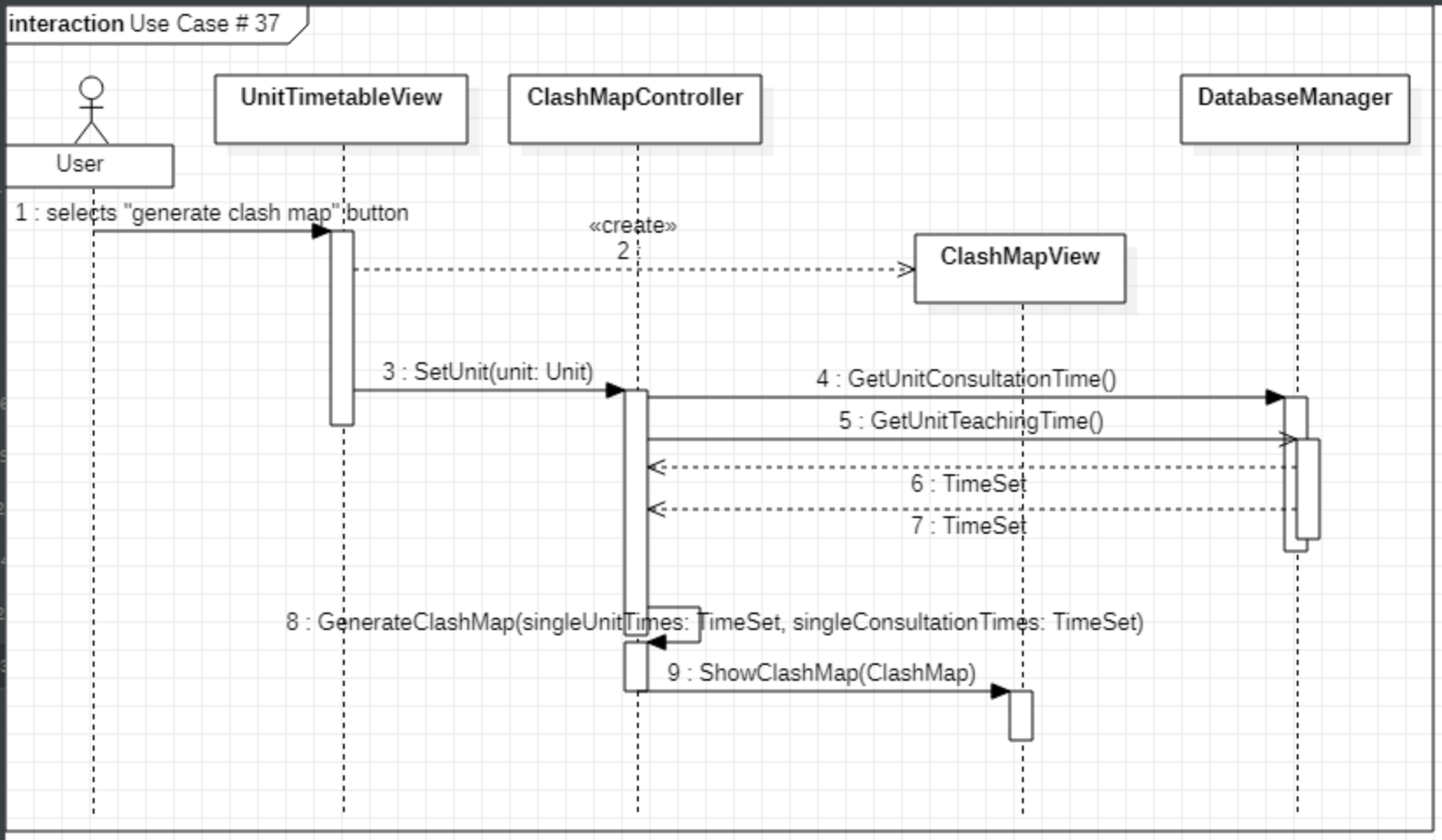

interaction UseCase#16
User
StaffListView
StaffController
DatabaseManager
1 : Selects a name
«create»
2 :
StaffDetailsView
3 : LoadStaffs()
4 : GetStaffDetails()
5 : Staff
6 : Staff
7 : ShowStaffDetails(staff: Staff)


interaction UseCase#19
User
StaffDetailsView
StaffController
DatabaseManager
1 : Selects "show activity" button
2 : GetStaffConsultationTimes(staff: Staff)
3 : GetConsultationTimes()
4 : TimeSet
5 : consultationTime
6 : GetStaffTeachingTimes(staff: Staff)
7 : GetTeachingTimes()
8 : teachingTime
9 : TimeSet
10 : ActivityGenerator(teachingTimeSet: TimeSet, consultationTimeSet: TimeSet)
11 : Activity
12 : ShowActivity()
13 : selects "show activity" again
14 : HideActivity()


interaction Use Case#23
User
MainView
UnitController
DatabaseManager
1 : Selects "Generate Unit List" Button«create»
2 :
UnitListView
3 : LoadUnits()
4 : GetUnitList()
5 : Units
6 : units
7 : ShowUnitList(unit: Unit)
8 : Presses "Enter" in filter-by-unit textbox
9 : FilterByUnit(filterText: string)
10 : List Updated


interaction Use Case#24
User
UnitListView
UnitController
StaffDetailsView
DatabaseManager
1 : Selects a unit
«create»
2 :
UnitTimetableView
4 : UnitClassDetails(unit: Unit)
3 : GetClassDetails()
6 : class
5 : Class
7 : ShowUnitTimetable(class: Class)
8 : Selects an option in filter-by-campus menu
9 : FilterByCampus(campus: Campus)
10 : GetStaffDetails()
11 : List Updated
12 : Staff
13 : selects a timetable entry
14 : SearchStaff(class: Class)
15 : ShowStaffDetails(staff: Staff)


interaction Use Case#28
User
MainView
HeatMapController
HeatMapView
DatabaseManager
1 : selects menu button "heatmap"
2 : ShowTimeOptionDropdown()
3 : selects time option
4 : SetHeatMapTimeOption(timeOption: TimeOption)
5 : ShowCampusOptionDropdown()
6 : selects campus option
7 : SetHeatMapCampus(campus: Campus)
8 : FetchAllUnitTimeSets()
9 : FetchAllConsultationTimeSets()
10 : Timeset
11 : TimeSet
12 : GenerateHeatMap(allUnitTimes: TimeSet, allConsultationTimes: TimeSet)
13 : ShowHeatMap(HeatMap)
14 : clicks "change primary colour"
15 : ShowColourPalettes()
16 : selects a colour
17 : SetPrimaryColour(colour: Colour)


interaction Use Case # 37
User
UnitTimetableView
ClashMapController
ClashMapView
DatabaseManager
1 : selects "generate clash map" button
«create»
2
3 : SetUnit(unit: Unit)
4 : GetUnitConsultationTime()
5 : GetUnitTeachingTime()
6 : TimeSet
7 : TimeSet
8 : GenerateClashMap(singleUnitTimes: TimeSet, singleConsultationTimes: TimeSet)
9 : ShowClashMap(ClashMap)

End of Document