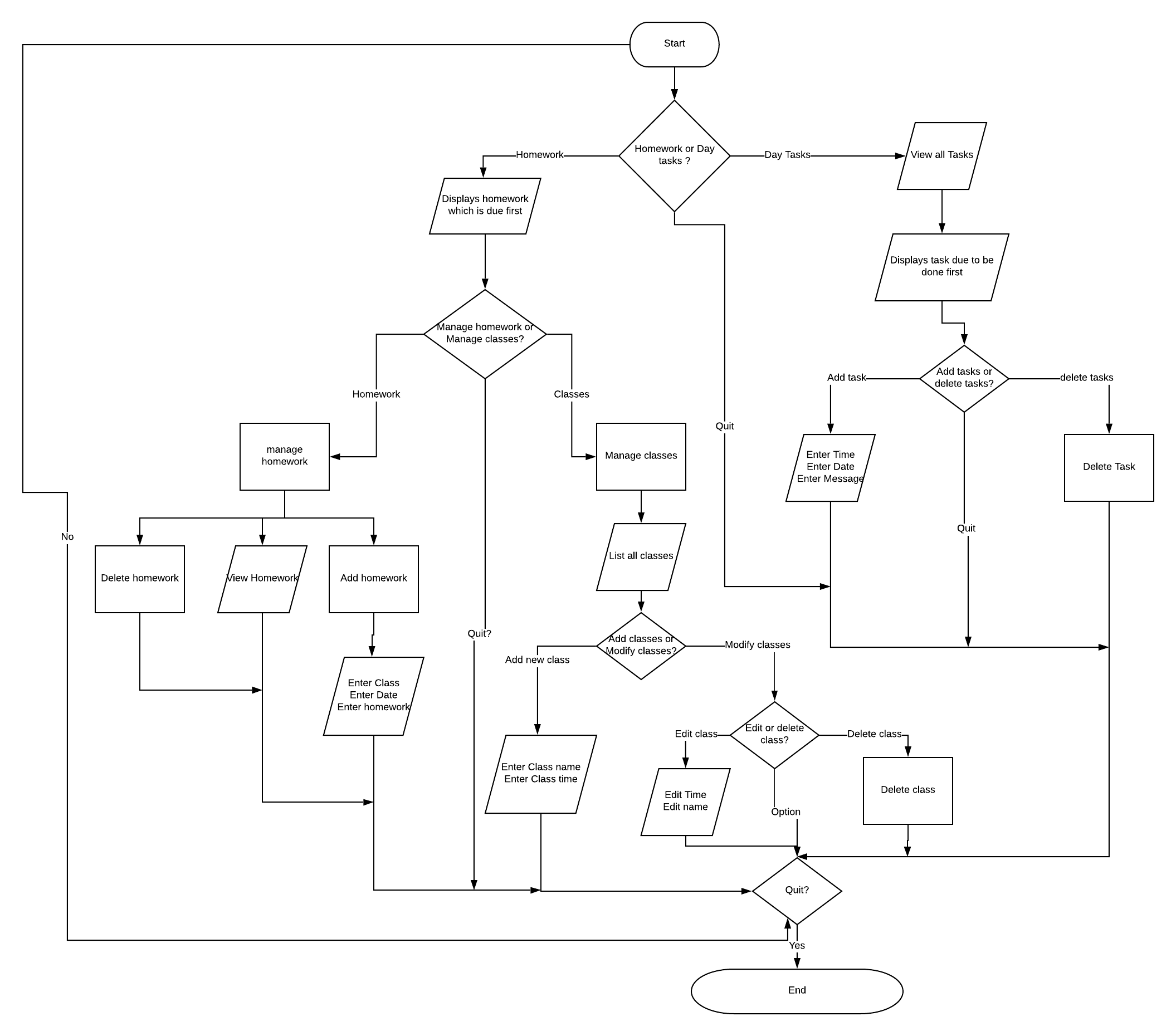
**Criteria B Design**



Flowchart of the flow of the program. How this is executed is shown in the UI design sketch that follows. Given that this is a phone application, specifically Android, the user can simply exit or click back at any time.

1. The user opens up the Main Menu Screen 1
2. decides which module to open between day tasks and homework.
3. The homework module shows the most urgent task then moves and lets them pick between managing classes or homework
4. Ether lets the user add, view and delete classes or add, view and delete homework. Adding either the homework or class opens a pop to take in all their fields.
5. On the other hand if the user picks Day Tasks at the first step then a screen displaying all the tasks and options to either delete or add new Day Tasks. A new Day Task can be added by using a pop up as well. The program can also be closed at any time or just go back on anytime.

The only full screens are :

Main Menu

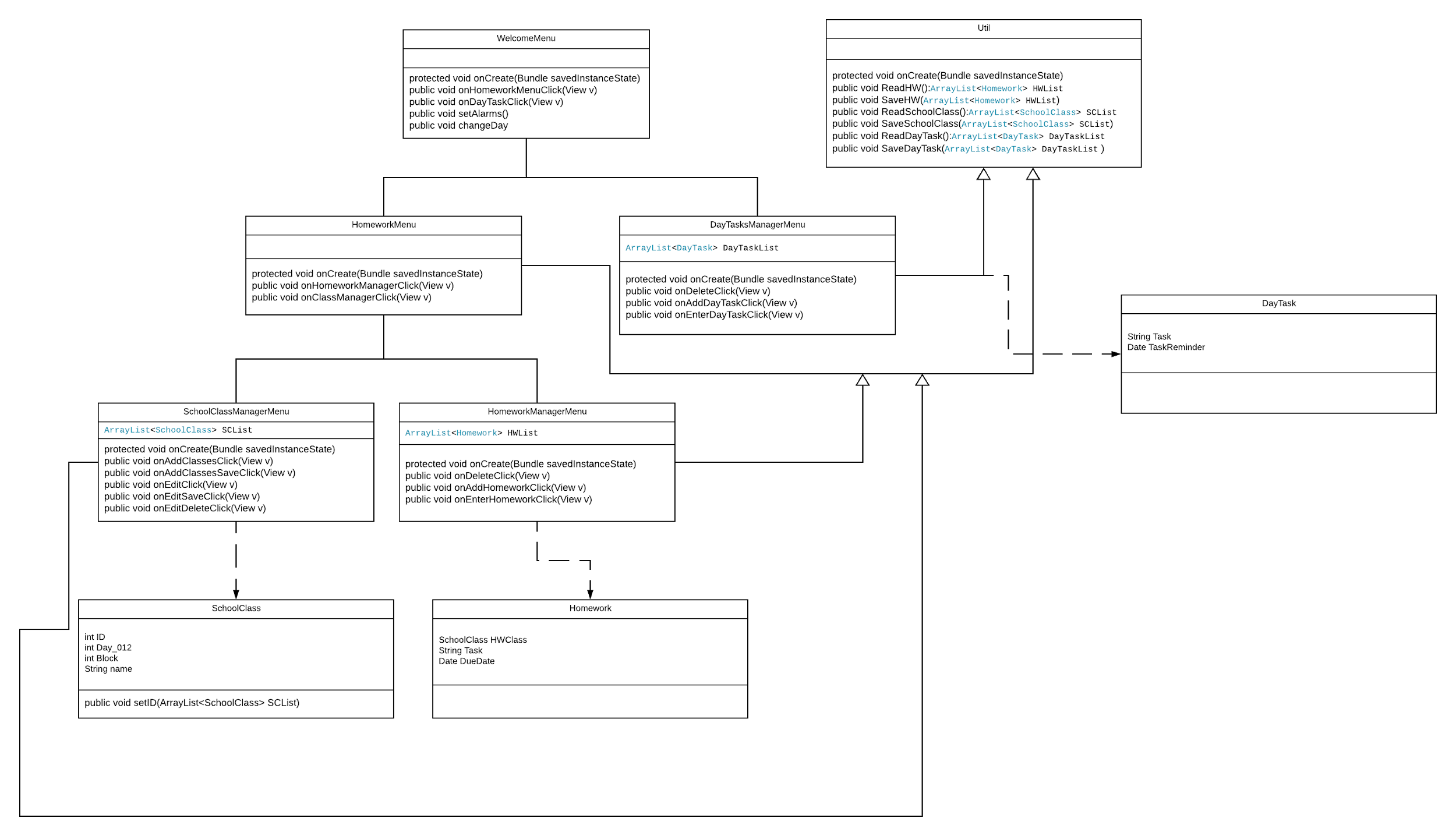
Homework Menu

Homework Display/Delete/Add Page

Class Display/Delete/Add Page

Day Task Display/Delete/Add Page

**UML diagram**

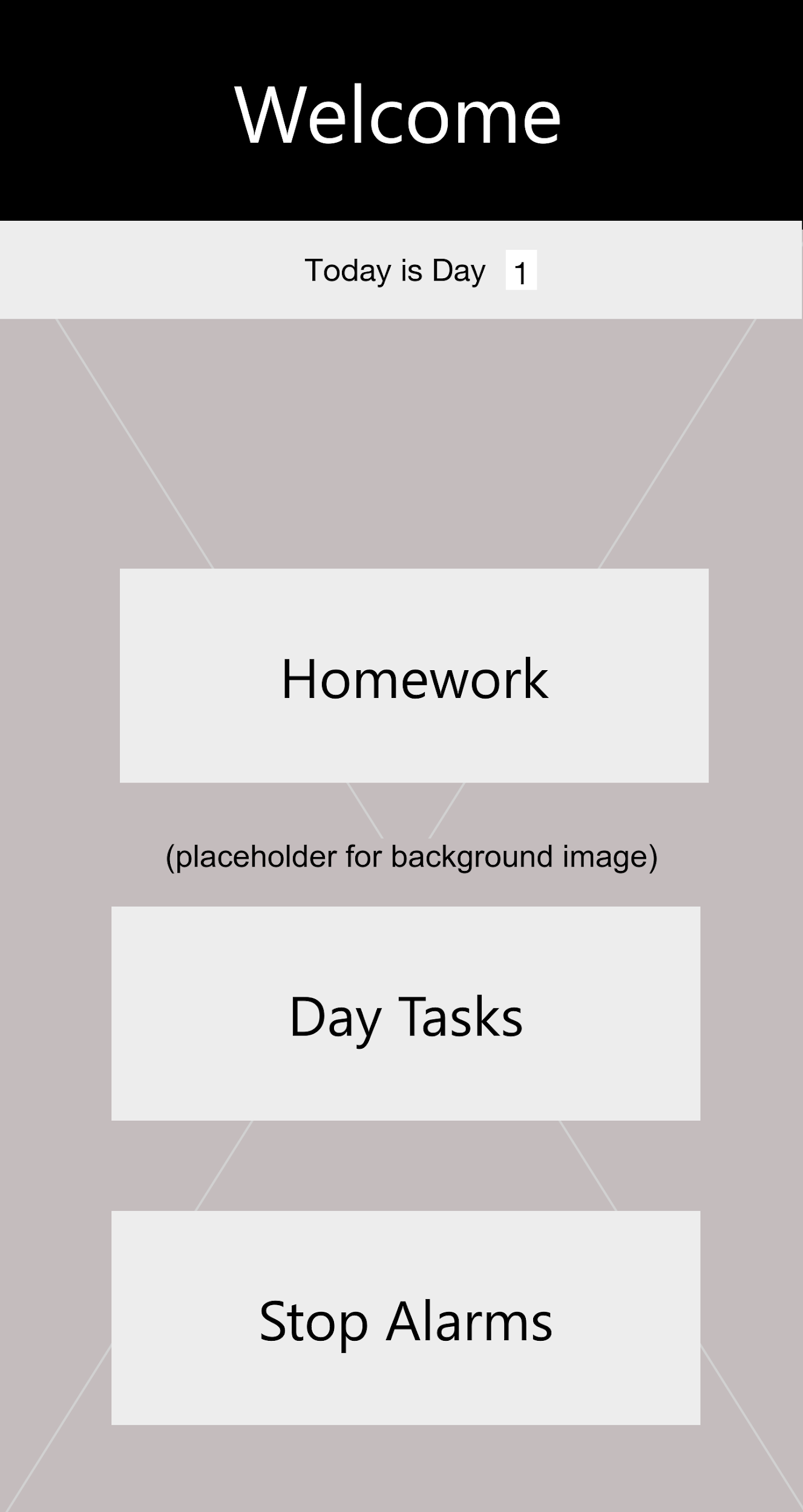


**In the UML diagram**

* Util class contains all the read and write of all files needed to be stored and is inherited by other GUI screens/Classes to setup each page
* HomeworkClass describes a single homework object and is data class with references to which Class it is assigned to, its duedate and a String for details
* SchoolClass represents a single class in school for each user containing a name, a unique ID, Day and Block information. It also has a SetID function to give it a unique ID when adding a new Schoolclass into the file or list.
* DayTask Class is another data class for Day Tasks and just stores the time and reminder.
* ArrayLists will be used as the variable type for each data classes because of its lack of a limit and get functions.
* HomeworkManageMenu will be the screen to display, add or edit all homework Items. Each onAdd,OnEnter and DeleteFunctions and the onCreate functions would use the util class methods for setup and functionality to constantly be updated and also keep the file updated. OnEnter happens when the new object is added while onAdd is method to refresh the page(GUI class)
* ClassManageMenu will be similar to HomeworkMangeMenu in functionality and code but instead display/add/delete School Class objects and hence use a SchoolClass ArrayList and related functions in util class for its setup, refresh, add and delete functions. (GUI class)
* DayTasksMangeMenu will be similar to the other ManageMenus is functionality, instead using DayTasks and its own methods. Setting the alarm will be handled through the onAdd and so on.(GUI Class)
* HomeworkMenu is a class to give the user option between ManageHomework and ManageClass. Each of its method corresponds to opening those pages (GUI Class)
* MainMenu is the starting page does the initial setup for the program. It sets the alarms, has functions for each new GUI to open up and another method to manage if the user wants to change the ‘Day’ (either 1 or 2 depending on which ‘Day’ it is) (GUI Class)

**UI design:**

Start Menu- Screen 1



**What does it do:**

* The start menu with 4 buttons
  + The Homework button leads to the

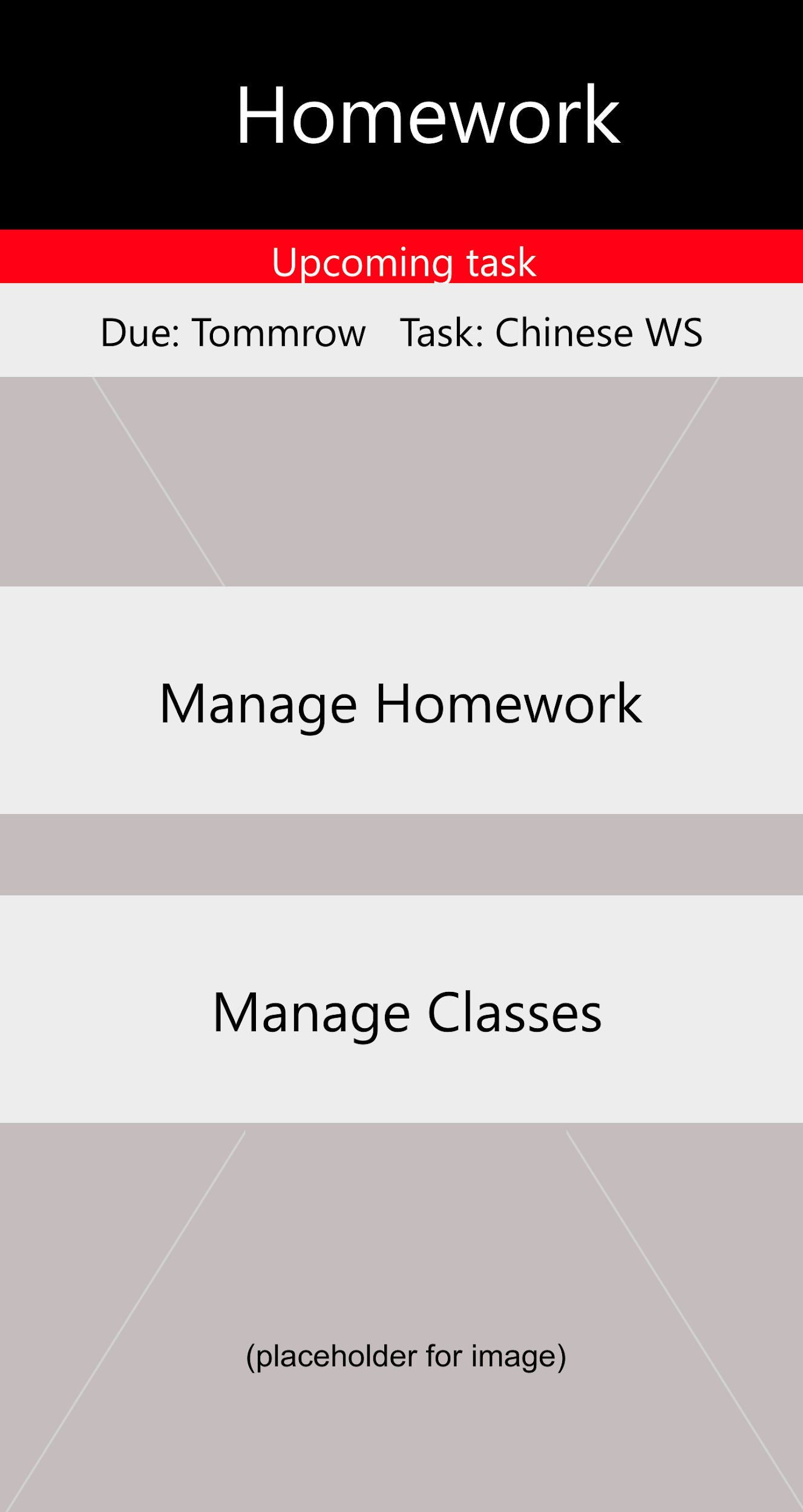
homework module section

* + The Day Task button leads to the Day Task module.
  + ‘Stop Alarms’ button is for holidays and/or any other issues.
  + The other option it has is to change the day in case of something unexpected like a holiday or extended vacation, done by adjusting the Day 1 or 2 in the textbox
  + This class sets up the alarms for classes

**Rationale:**

When using the app, the user should not have to dig to get to the core features. If I have to enter a homework or class, I would not need to enter an alarm, and vice versa so the distinction between going into either module is available. The app only has two manual settings needed to manage the alarm system and, for easy access, they’re straight up available as well because if the user intends to edit just them then they shouldn’t have to go through the rest of the program. A simplistic design suits the requirement and unnecessary fluff is not included to delay the user since there was no requirement for additional features from the user and the UI needs to be custom fit to meet the demands of being a school app being as minimal as possible.

Upon clicking the Homework Button:

Homework Menu- Screen 2

**What does it do:**

* Shows upcoming task and how many days left. A red line above for extra urgency
* Mange homework opens a page to view, edit, add and delete homework tasks
* Mange class opens a page to edit, view, add or delete classes

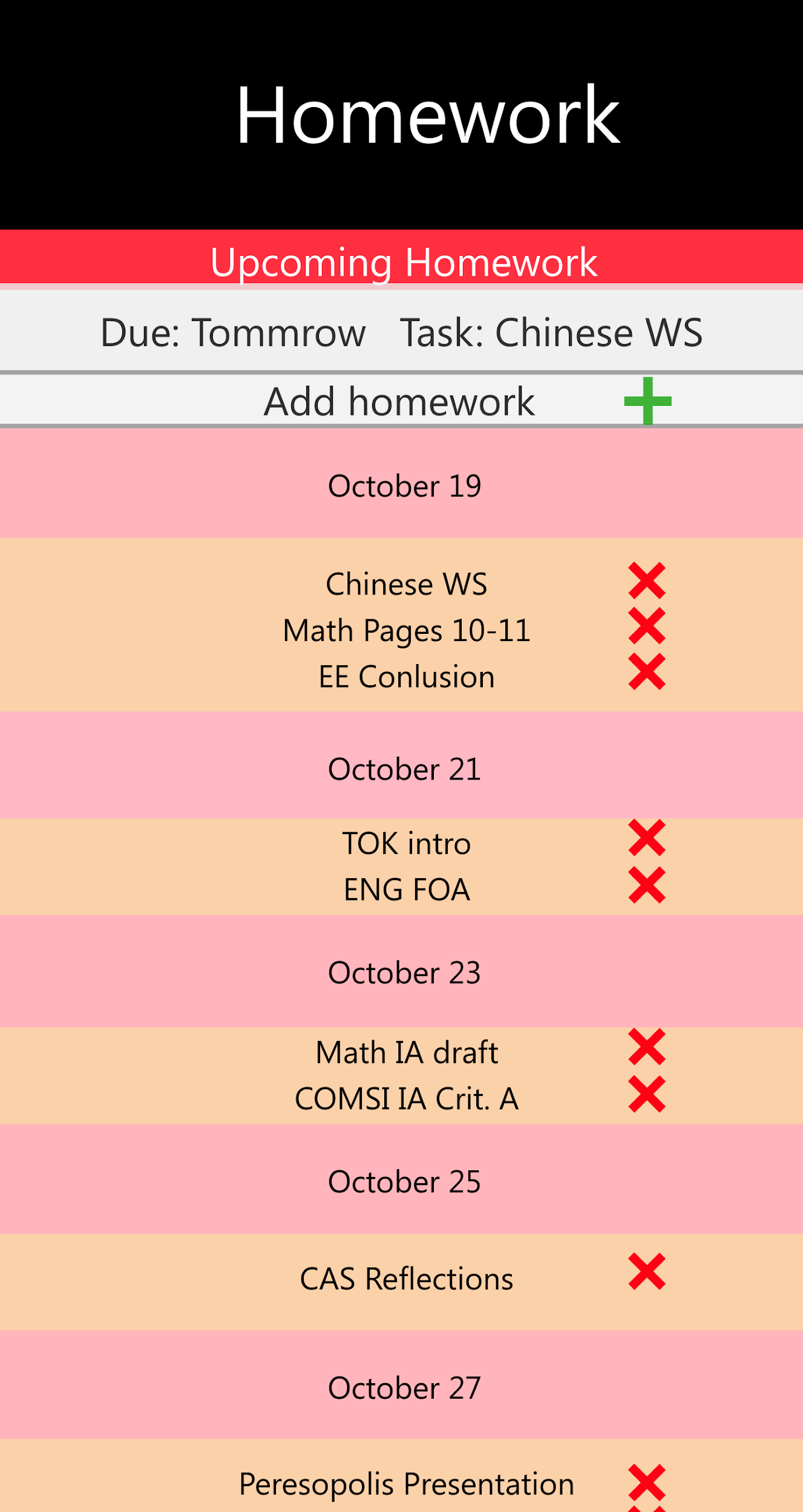
**Rationale:**

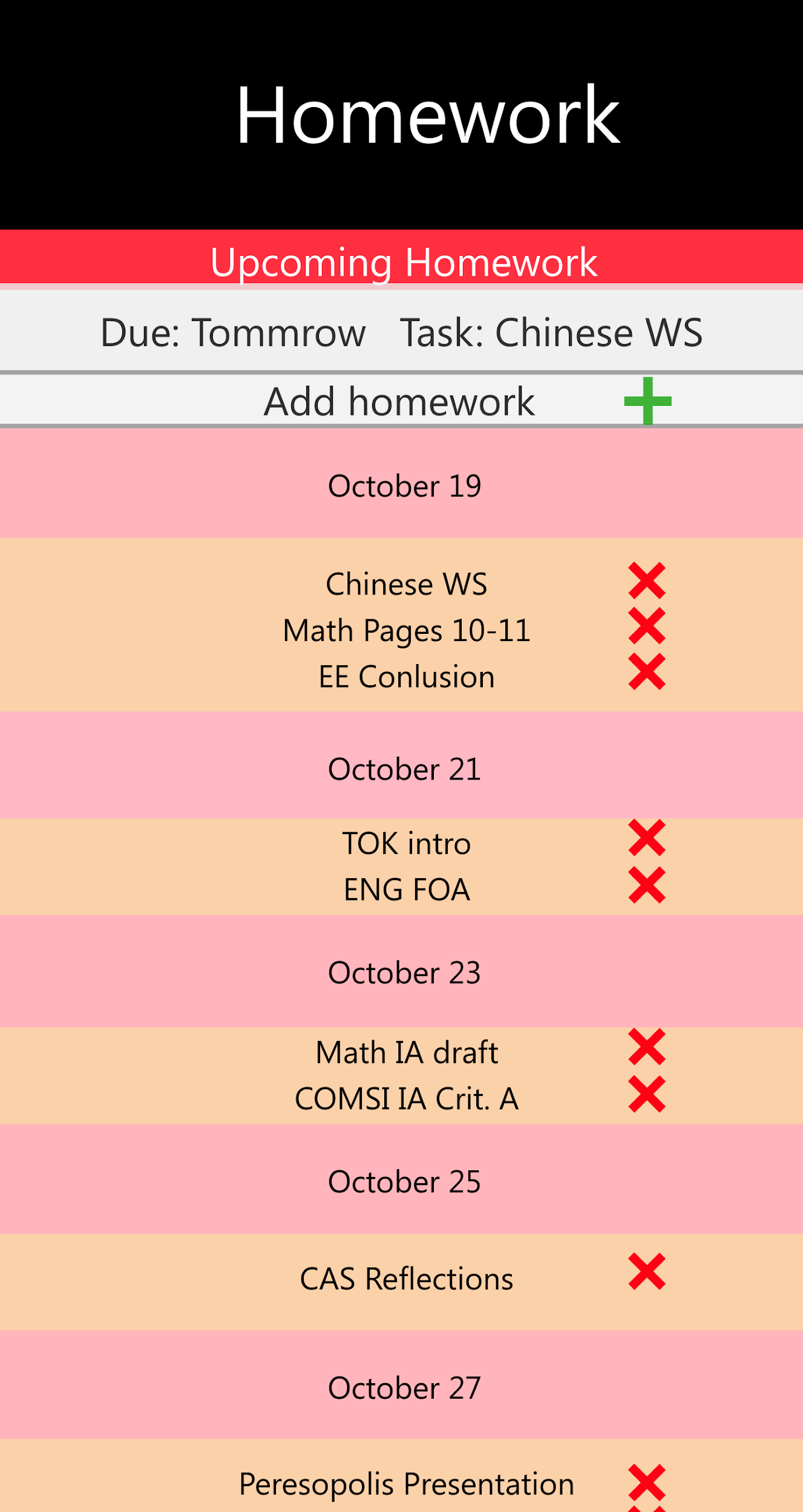
The reason why the button for classes is on this screen and not screen 1 is because too many buttons on screen 1 would clutter the screen and only 2 buttons let the user quickly read then tap to the next menu. Since fewer options means the less text to read and less overwhelming to the user. And the style of two buttons per page remains consistent. If the user wants to do something in the homework related side, small quality improvement is to display the most recent homework since that is the issue in criterion A, a small bit to help adds extra polish. From here the choice between homework and class is important since both are focused on their sections only and combining both of them into one screen would lead to too much clutter and irrelevant information if the user just wants to edit/add/etc.. only one of them.

.

Upon choosing Manage homework from that screen:

Homework Management- Screen 3





**What does it do:**

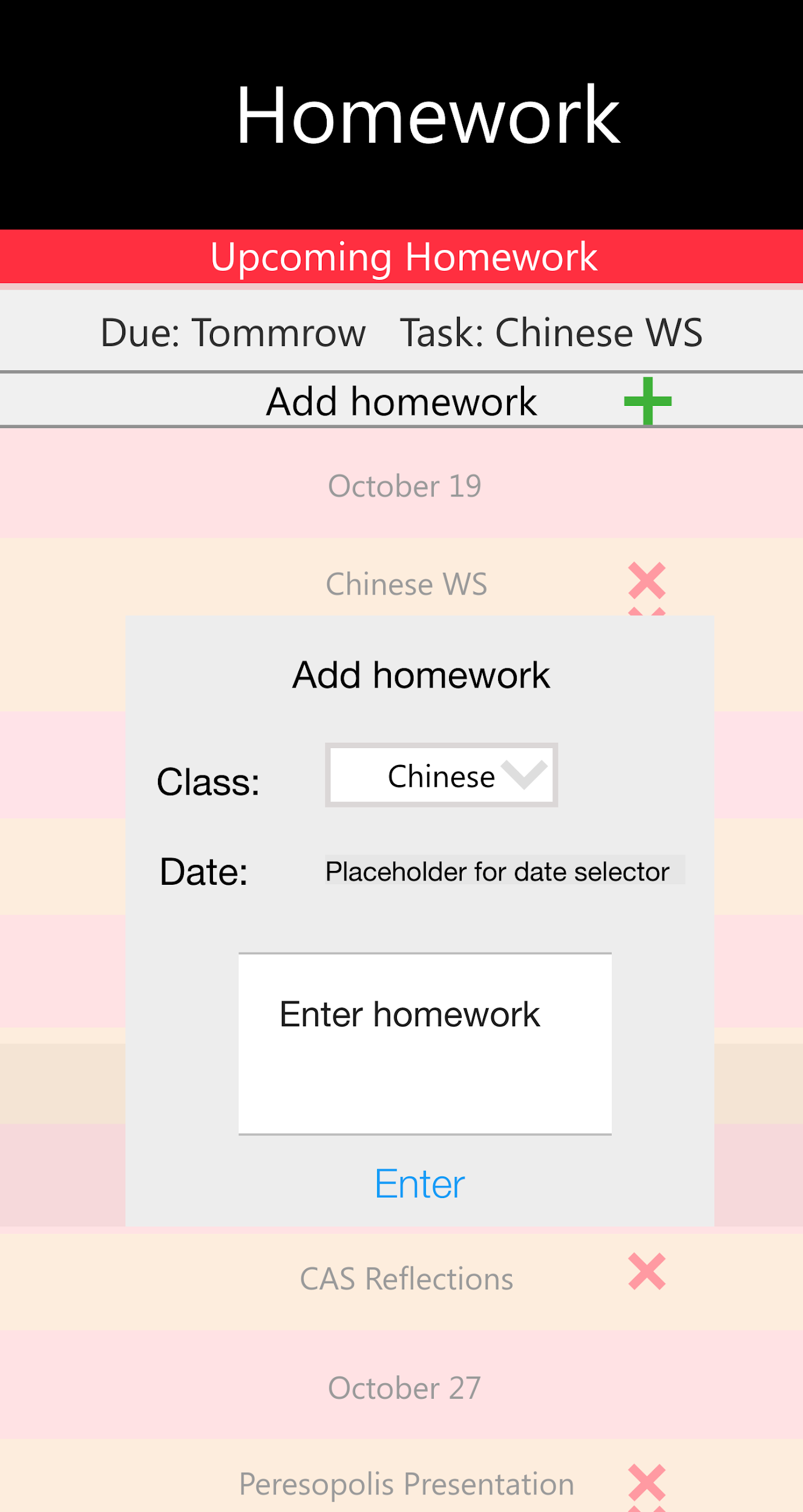
* The red X is a button to delete the task
* The + button opens a pop up to let the user add a homework task
* All of the user’s homework is displayed sorted by closest deadline and in a standardized format of ‘Class--task’ underneath the due date

**Rationale:**

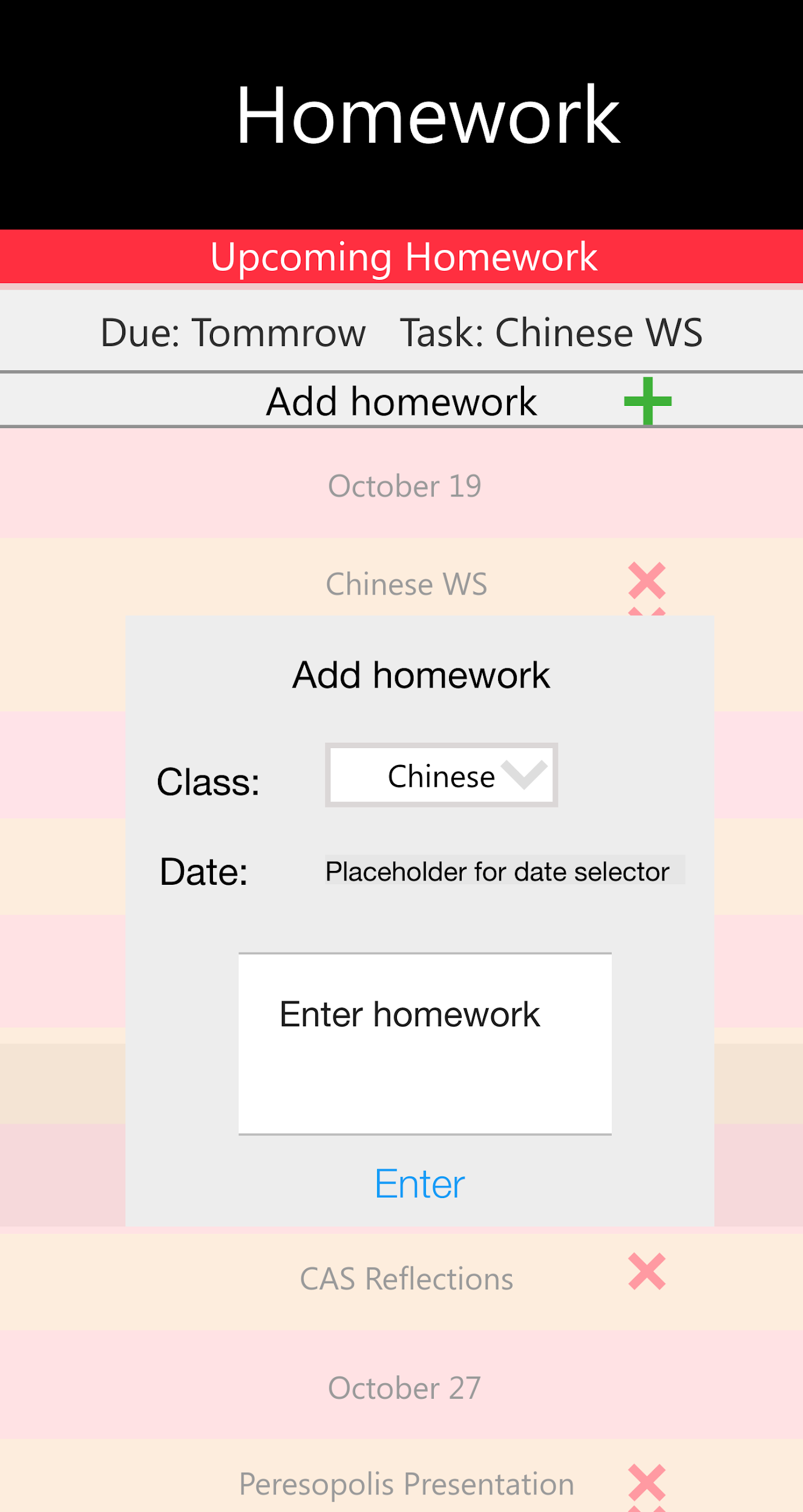
Because the expectation is the user will delete and add homework tasks multiple times daily, so they only need to do one action(click on X), and it is done by using a recognizable symbol and color to make it easy to understand its purpose. The X is also a bit offset to the right because when normally using phones, the user’s fingers are usually at the center by natural habit, by putting the X to the right prevents the chances of accidental deletion. The homework is displayed in a scrollable list because the other implementation would be to have the user side swipe through screens which would be uncomfortable. A vertically scrollable list is a common feature of phone apps and is the best solution to match android phones. The list is chronological so the most important items are always first visible to the user and the farthest due are at the bottom so the focus in on the immediate. The plus button is another universal symbol and is a at top so it isnt accidently pressed but also clearly visible. The most recent task is no longer displayed since it is already in the list. The colors are prone to change but dates and homework tasks will have a different color so it is always clear when it is a different day.

Upon clicking add homework

Homework Management- Screen 3.1



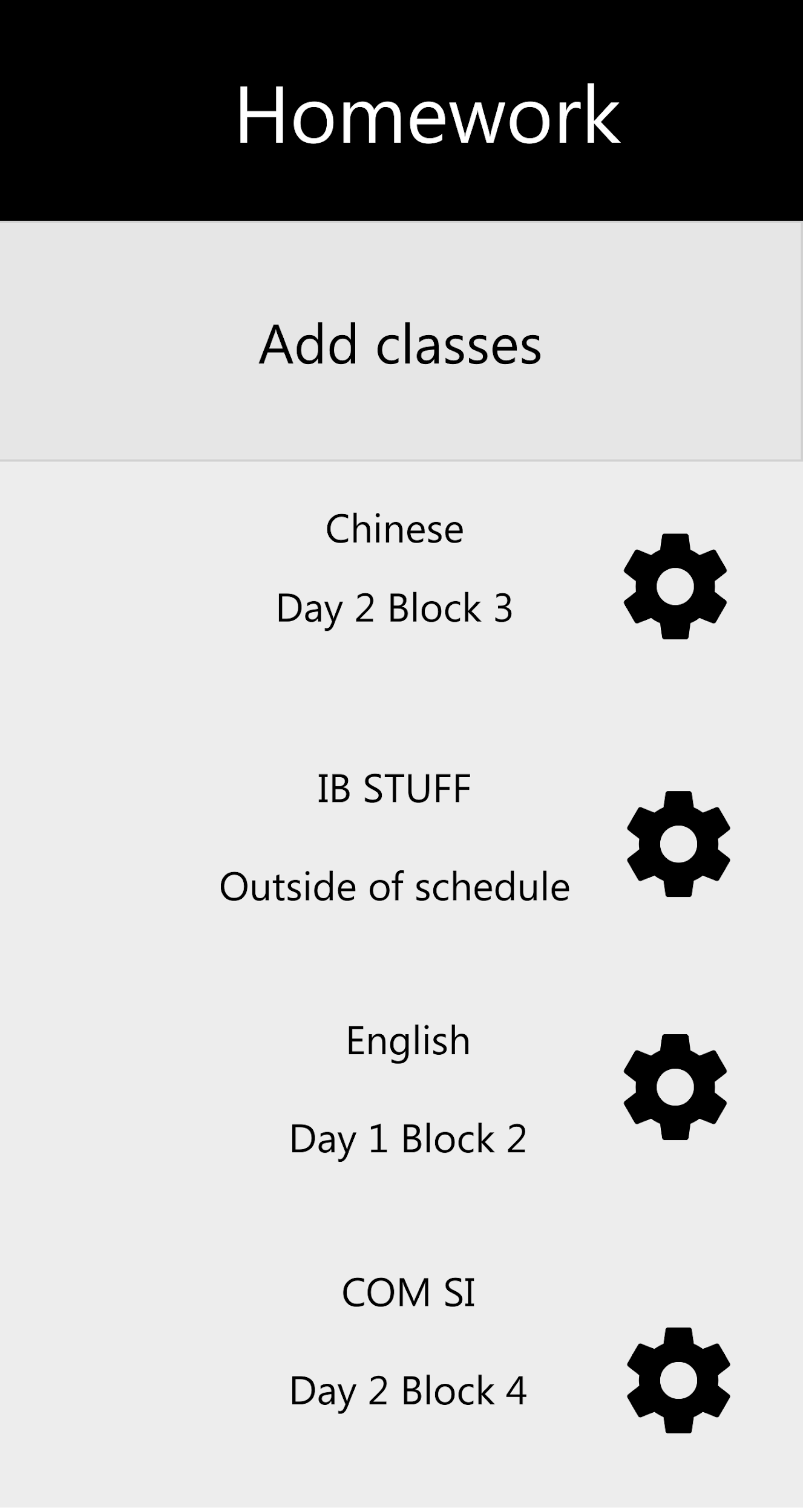
**What does it do:**

* Opens a pop menu to add a homework task
* Classes are in a drop down list
* Date would be selected via a mini calendar
* A text box to input homework details
* A enter button to add the homework

**Rationale:**  
A pop menu is easier than a whole new screen since only a few fields need to be filled. Use of drop down list and a mini calendar makes it easier for user to input data(compared to manually putting in strings) and also validates since those two methods can’t be incorrect and are easily understood. The text box accepts string input and is valid automatically since all of the phone keyboard inputs are accepted. The enter button makes the user consciously aware about inputting all their data and is a check that they are satisfied with it.

Going back to Screen 2, Homework Menu screen, if the user clicks Manage classes instead, this screen appears:

Class Management- Screen 4



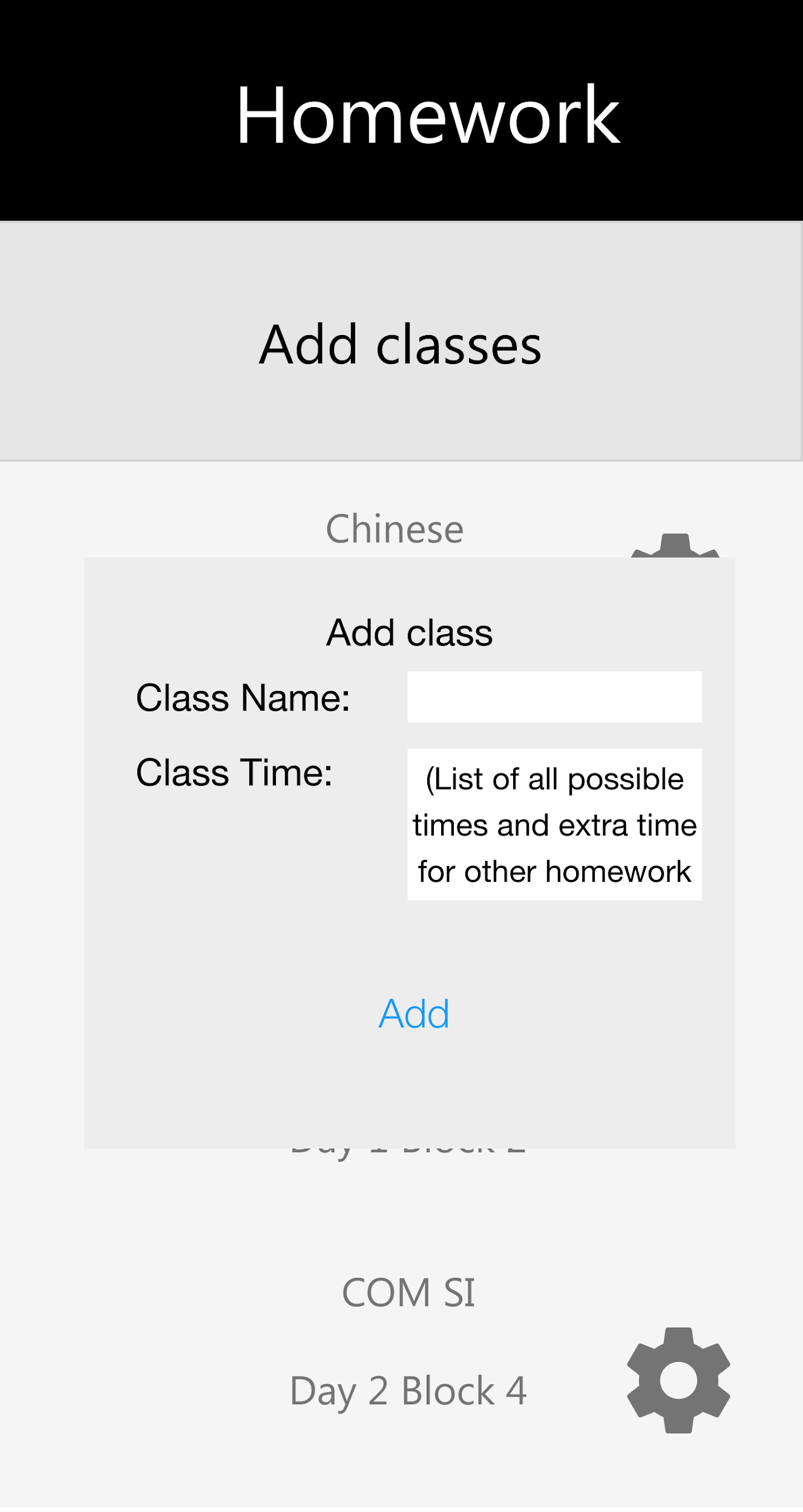
**What does it do:**

* Display all user added classes in a list
* Add classes button opens a pop up to add a class
* The gear cog is a button to open a pop up to edit or delete the School Class next to it

**Rationale:**

The rationale for vertical lists is already provided in the rationale for screen 3. The gear cog is a widely recognized symbol that represents settings and in this case the setting of each class. For ease, the time and name of the classes are also already shown to minimize number of menus. The add class button is similarly positioned to the add homework button in screen 3 though, a bit bigger since the number of classes is expected to be less. Since there is extra space, a + symbol didnt seem needed and just text was enough.

This screen is what happens when the user clicks the “add classes” button.

Class Management Add classes- Screen 4.1

**What does it do:**

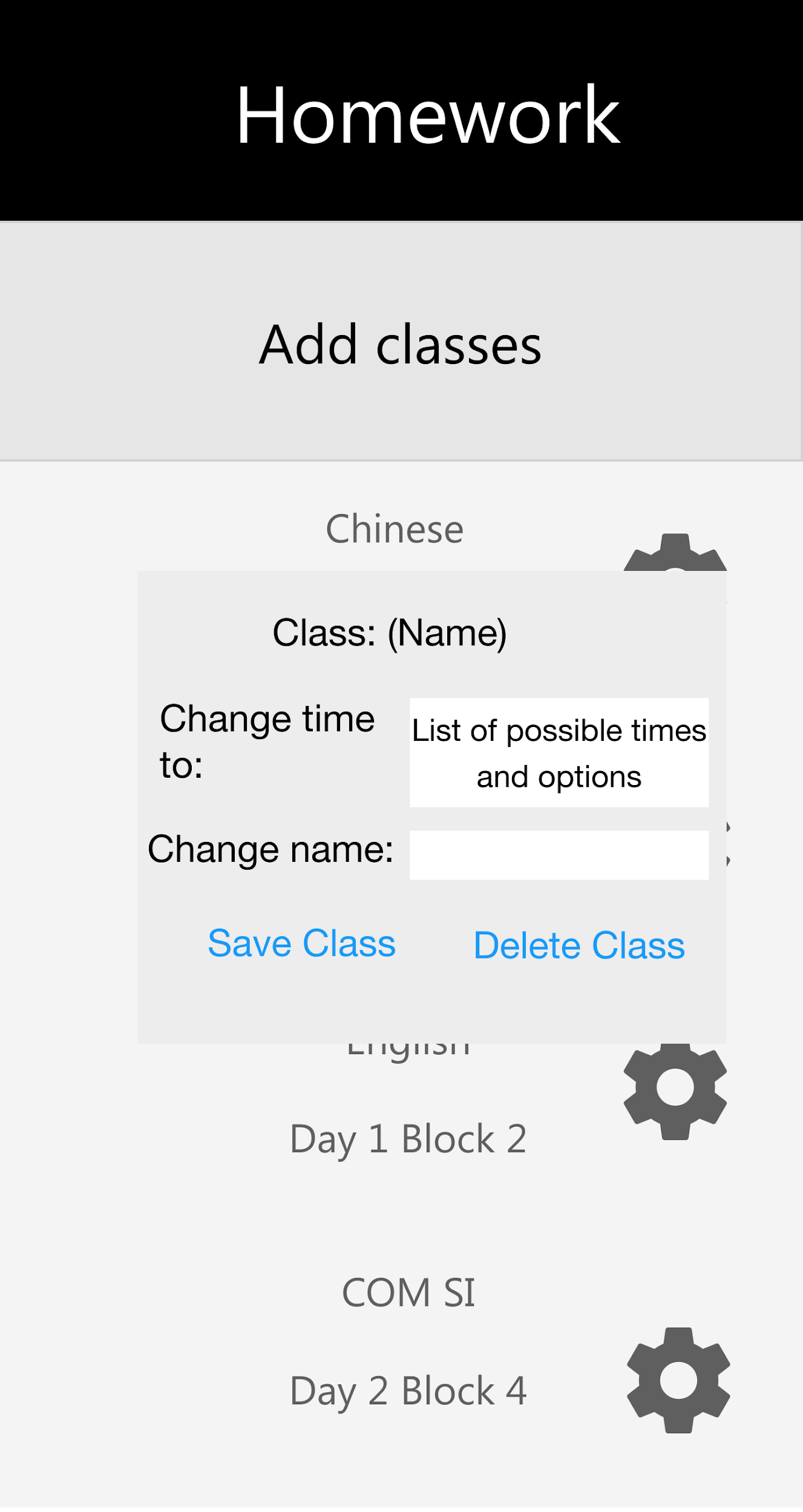
* Open popup menu to let the user add class
* The class name is a string input
* The class time input is a drop-down list of all possible times for classes, ranging from blocks 1-4 for days 1 and 2 and a separate option for unscheduled

**Rationale:**

A popup menu is used again since only a few fields are needed. The class name can be anything hence a string input via the phone keyboard. The class time is a drop-down list to add validation and because the times are preset and should be easy to setup like stated in criterion A. The class time also includes an option for unscheduled for work that isnt linked to any class, the app cannot set reminders for that but can help in organizing that data and lets the app be a single solution to the homework issue and hence, including unscheduled is an extension of that.

This screen is what happens when the gear cog is clicked.

Class Management Edit Classes- Screen 4.2



**What does it do:**

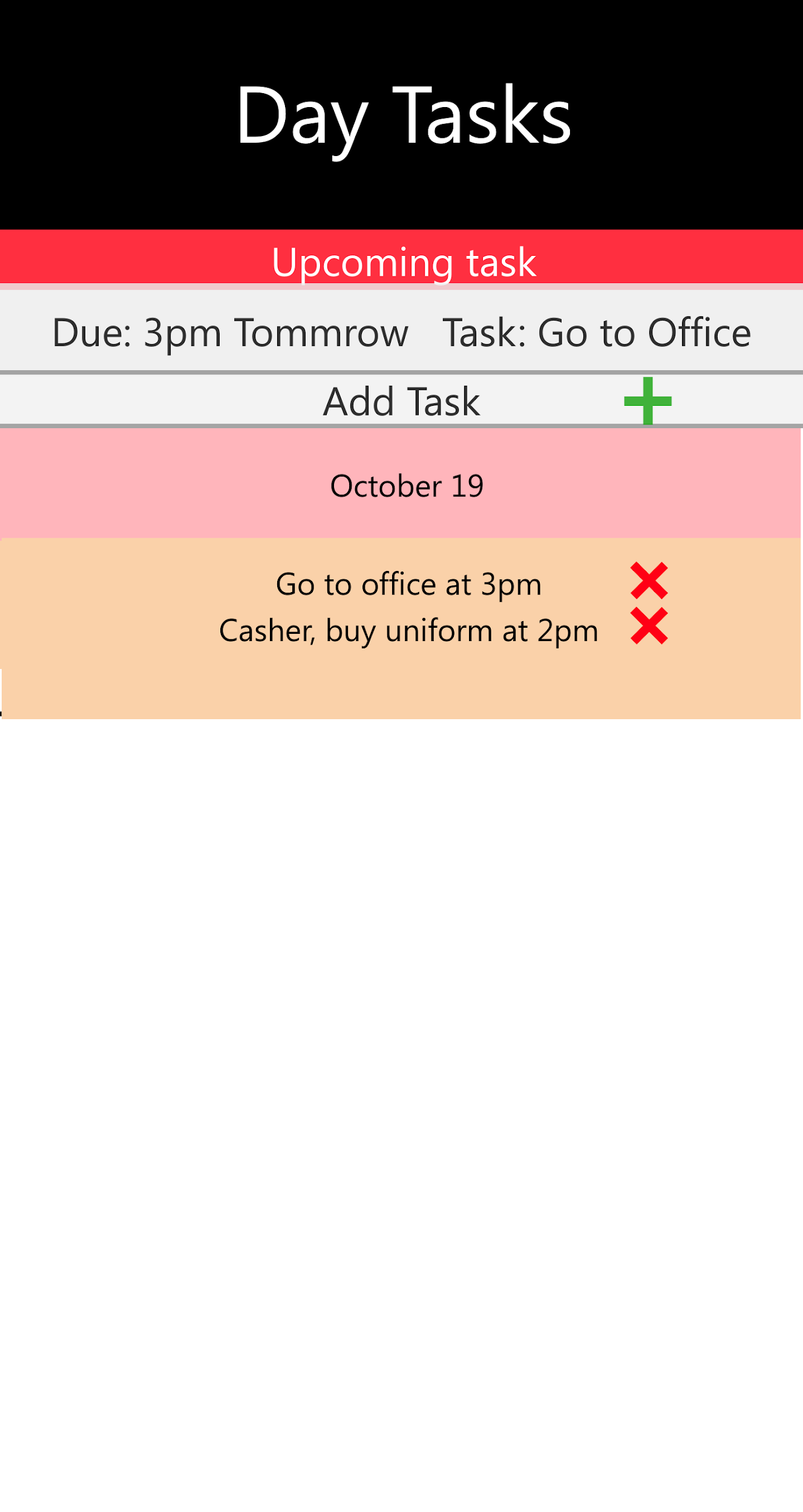
* Open a popup titled the Name of the class being edited
* ‘Change time’ is a drop down list containing all the possible times, the default time selected will the pre existing time of the class being edited
* Change name is a text box that will by default hold the pre-existing name of the class
* Save class saves the class with the edited fields
* The delete class deletes it from the list of classes and the file

**Rationale:**

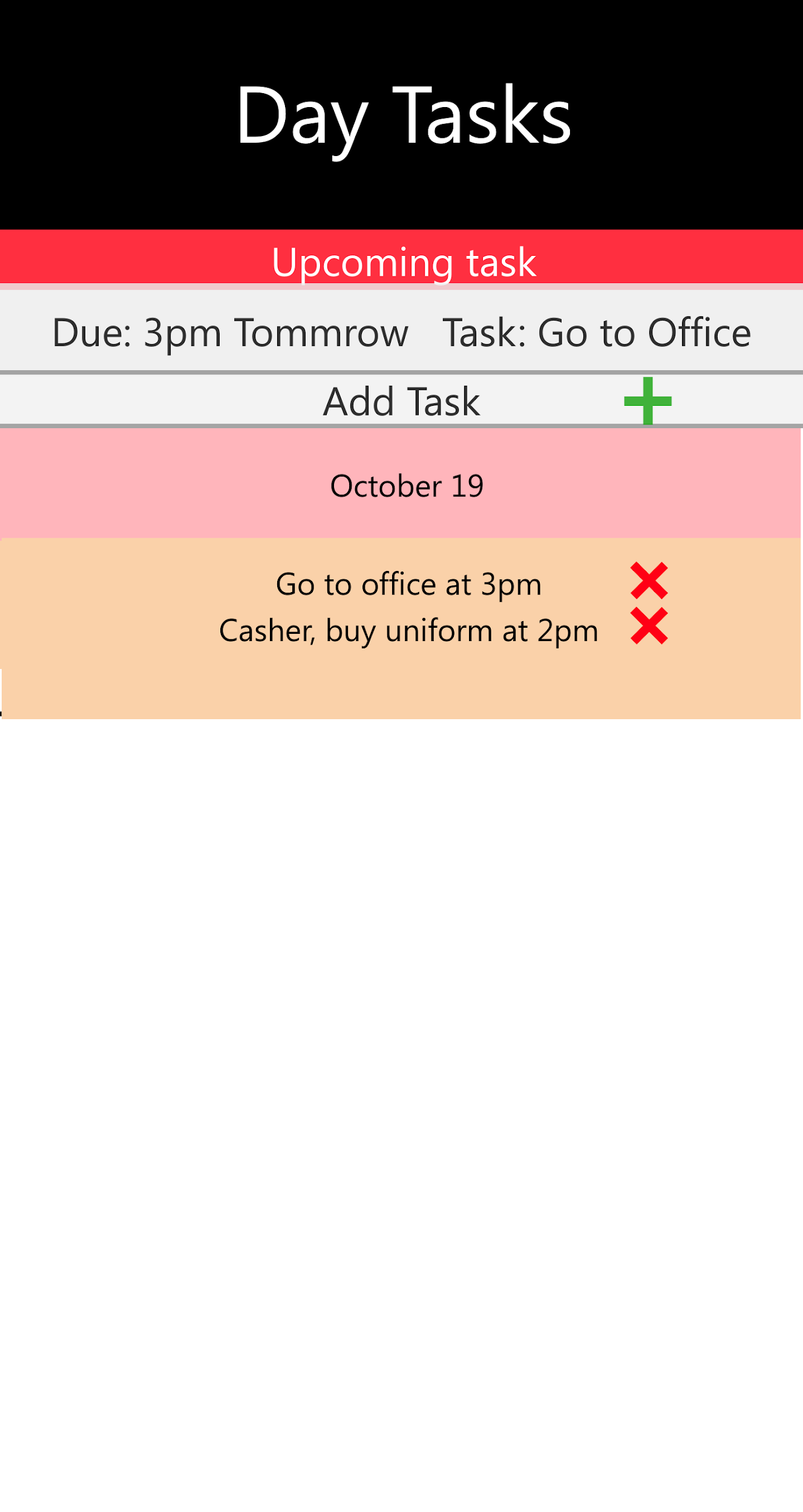
The name of the class is there so the user doesn’t forget the class that is being edited. The selection of the drop down list by default is of the pre-existing data belonging to the school class being edited maximizes efficiency so the user doesn’t have to re-write or re-select the time if they only want to change one of the fields. The save class makes the user consciously acknowledge their changes before saving them to minimize mistakes. The reason delete class is on this popup and not screen 4 is because deleting a class has more implications than deleting a homework task. If a class is deleted all the associated homework tasks will also be deleted. To prevent accidents, the user needs to perform an additional step here. Also, the user is not expected to regularly delete classes so it’s accessibility is not a serious concern.

Going back to screen 1, if the user picks Day Tasks instead, the following screen appears:

Day Tasks- Screen 5



**What does it do:**

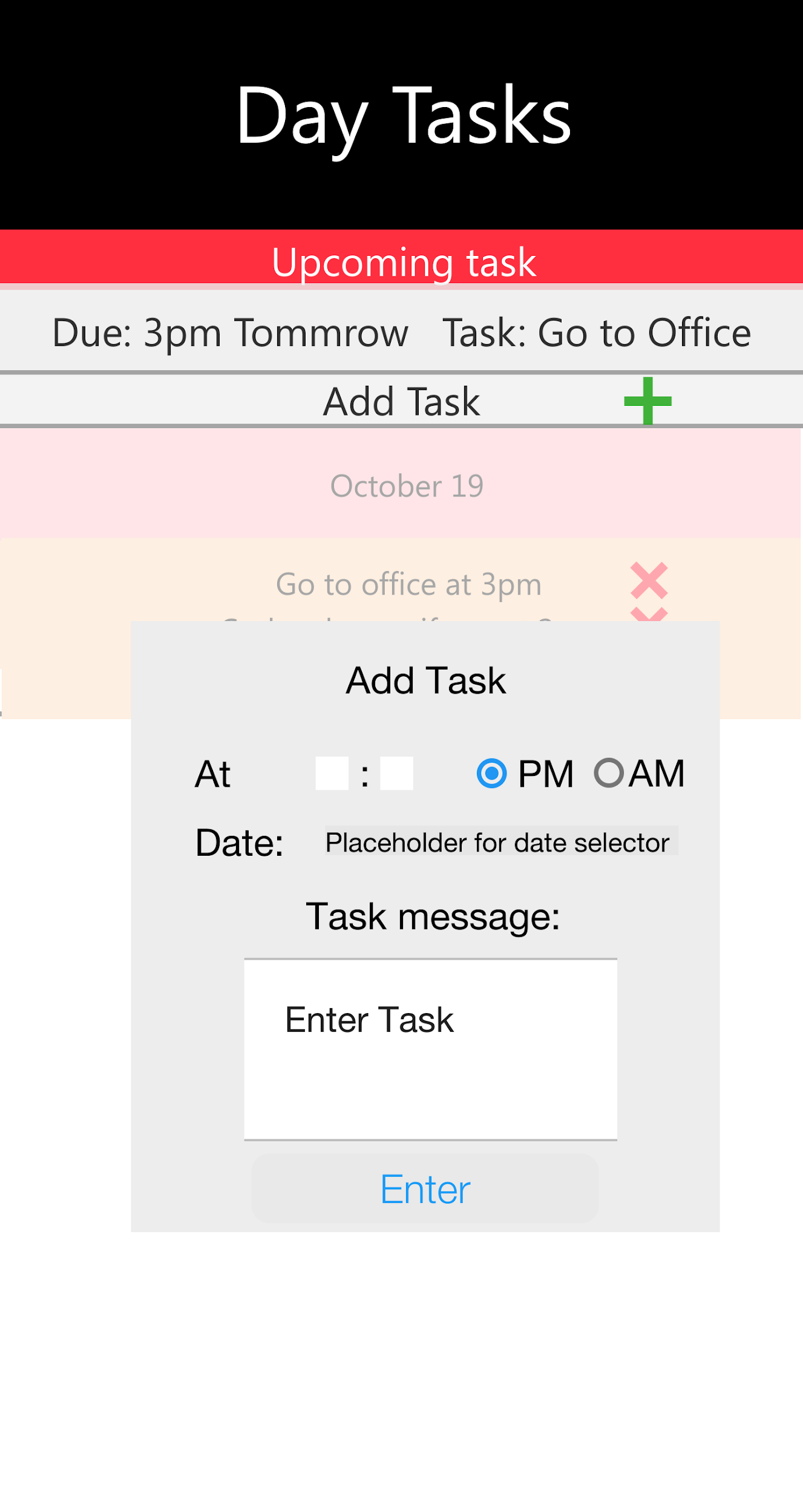
* Shows all set day tasks
* The X is a button to delete the associated task to it on the right
* The most urgent task is displayed in red headline above it with the task detail and time
* The tasks are chronologically sorted by dates with different colors for dates and tasks
* The Plus button opens a popup to add a new day task

(Note, the bottom half the screen is blank due to not being more tasks in this mock screen)

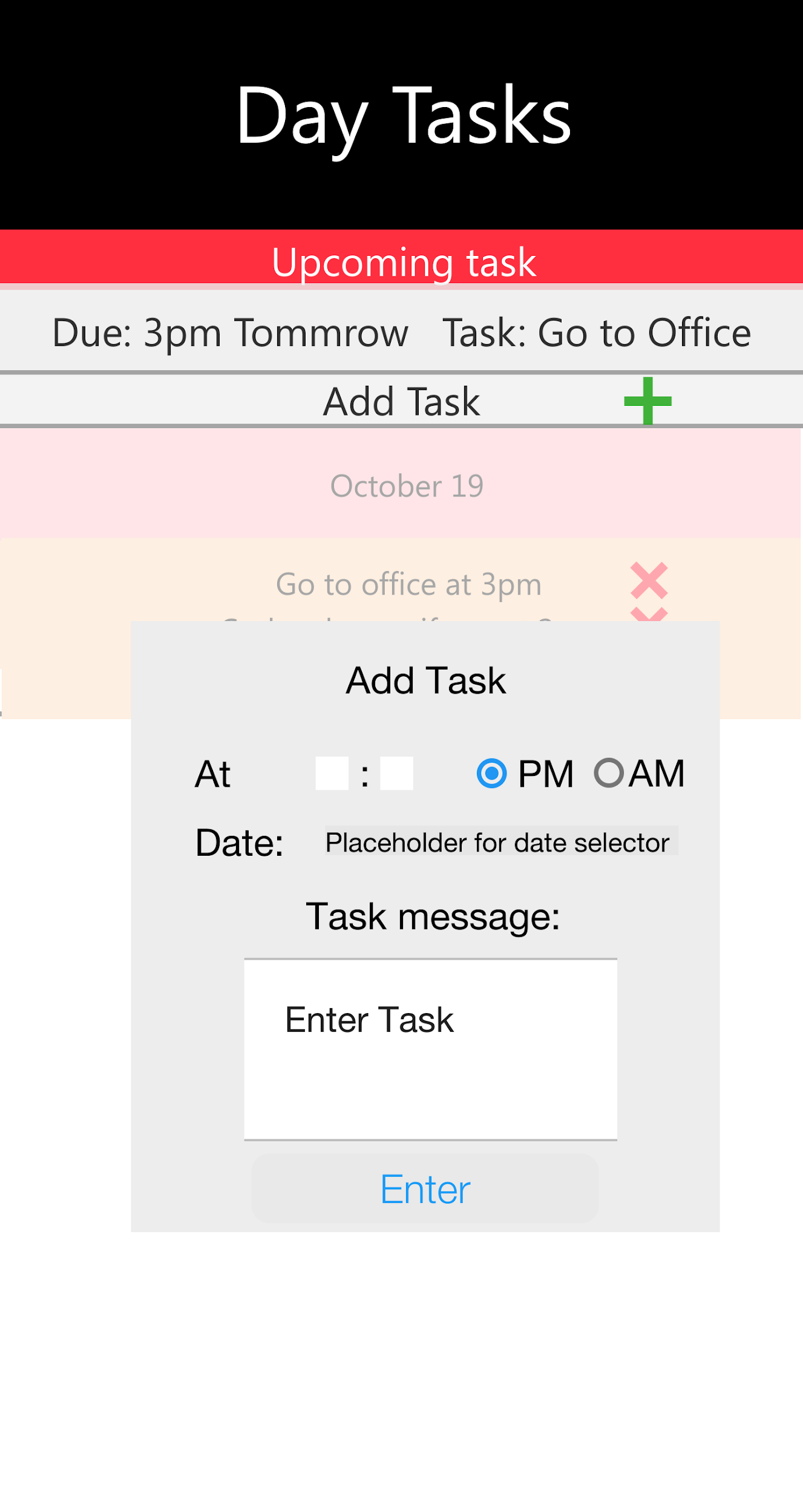
**Rationale:**

The overall design is similar to Screen 3 of the homework to keep consistency since the purpose is similar, to display, add or delete data. The use of a green + and a red X is a recognizable symbol and doesn’t require additional explaining. Clicking on X would delete the task., because the tasks would be deleted frequently they do not require a validation but instead a single click only. Again the cross is easily recognizable and offset to the right, because by default when using a phone fingers tend to stay near the center or very close to the edge. The placement of the X is just far enough from both to prevent accidental presses. The + button is universally understood to add and is separated enough that it won’t be accidently clicked. The tasks are shown chronologically so the user is always aware of what is relevant.

Click on Add Task opens this screen:

Day Tasks- Screen 5.1

**What does it do:**

* Open a popup
* Enter a date using a calendar
* Enter AM or PM via radio button
* Enter Hours and Minutes using a drop down scrollable list
* Enter task via textbox
* Enter button to add the task to the list on the previous page, save it and an alarm.

**Rationale:**

A popup is used again to keep it consistent with all the other add functions. The date is entered via calendar so the user cannot input something to cause an error and the calendar is also easier to use and look at because of the context it gives with each date such as day and date. AM and PM is a radio button to prevent the user from mis-typing and is simpler to use over entering something like 13:05. The use of drop down list for hours and minutes also prevents the user from entering wrong inputs that could cause error. The use of drop down would be uncomfortable for minutes since it has 60 options. To address this, making it scrollable makes it more bearable since its extremely easy to scroll to the rough area of what exact minute is wanted and a familiar implementation from the default alarm application in android. The rationale for Textbox and use of an enter button is same as for previous rationales.

Final UI notes:

The UI design is a rough sketch of the UI of the final diagram, however, the following things were not added; A help button on each page to guide the user with instructions. This was omitted because in a rough sketch it was unnecessary and can be added at a later date in development with more finalized structures. The date selecting via calendar was omitted for simplicity's sake and instead, a placeholder was then used to refer to it. The date would be done by choosing a date on a mini calendar. A back button was also omitted because the client’s device is an Android phone and has a back button but may be considered for later design. With the use of that button, the user can execute the back function easier than it would have been with an on-screen button. In the final UI, the font, color, and layout are subject to change, however, this will be the template for it.

**Plan for testing**

|  |  |
| --- | --- |
| Test Subject | Method |
| Test self-automation of School classes | Create a (school) class and check over a period of 2 days for self-automated alarms that provide a notification |
| Test self-automation of Day tasks class | Create a task and check if the next day at the correct time and day an alarm is sent and played by the phone. |
| Overall design | Have it tried out by various people and ask if they think there is extra buttons or screens and if so, where and any suggestions and iterate multiple times. |
| Test if data is being stored correctly | Create multiple classes, input homework for each, restart the phone and check if all the data is correctly read and displayed |
| Test efficiency of setting up homework and entering day tasks | Have the application tested by others and ask about their experience and see if they have any suggestions, anything they disliked and liked. |
| Test to see if classes and homework can be added and deleted | Create a class, add homework to it, delete the homework, change the time of the class, test if the alarms work then delete the class and see if the alarms stop |
| Urgent deadline function | Create a homework item and create another homework but due before the previous one and check if it updated properly |
| Chronological order homework is displayed | Put in multiple different homework items at various dates and compare dates afterwards |

Words: 2884