**UC01 - Create Task**

**Actors:** Creator (primary), CrowdSourcer

**Goal:** After a creator completes *Create Task* a new task will have been saved into the system.

**Trigger:** Creator has selected the "Create Task" button from the main screen.

**Pre-conditions:**

* The system is ready to receive a new task.

**Basic Flow:**

*Create New Task*

1. The creator enters the title into the system.
2. The creator enters their name into the system.
3. The creator enters a description into the system. (optional)
4. The creator enters the required number of items for the task into the system.
5. The creator checks whether the task is shared or not with other contributors into the system.
6. The creator submits the newly completed task to the system.
7. The system adds the current date and time as the creation time to the task.
8. The system adds the devices Android ID as the Creator ID to the task.
9. The creator is notified by the system that the task was created successfully.
10. The creator is transitioned back to the main application screen by the system.

**Variations:**

*Abort Task Creation*

1. If at any point before 5. in *Create New Task* and the creator clicks the "Cancel" button.
2. The creator is transitioned back to the main application screen by the system.

*Create Shared Task*

1. If at 6. in *Create New Task* and the creator has chosen to share the task.
2. The system uploads the new task to CrowdSourcer.
3. The system saves the newly created task into the list of shared tasks.
4. The use case resumes back at 7. in *Create New Task.*

*Create Local Task*

1. If at 6. in *Create New Task* and the creator has chosen NOT to share the task.
2. The system appends the new task to the local task file in memory.
3. The system saves the newly created task into the list of local tasks.
4. The use case resumes back at 7. in *Create New Task.*

**Post-conditions:**

* The creator has successfully created a task into the system

**Exceptions:**

*Missing Title or Creator Name*

1. If at 5. in *Create New Task* and the creator has not filled in either the task's title or their name.
2. The creator is prompted by the system which information is missing.
3. The use case resumes back at 1. in *Create New Task.*

*Incorrect Number of Task Items*

1. If at 5. in *Create New Task* and the creator has not entered an integer that is greater than or equal to 1 for the number of task items.
2. The creator is prompted by the system that the creator has entered an incorrect number of task items.
3. The use case resumes back at 1. in *Create New Task.*

*Cannot Upload to CrowdSourcer*

1. If at 2. in *Create Shared Task* and the system was unable to upload the new task to CrowdSourcer
2. The creator is prompted by the system was unable to upload the task to CrowdSourcer and that the task is being saved locally instead.
3. The system changes the task to NOT shared.
4. The use case jumps to 2. in *Create Local Task.*

*Cannot Append Task to Local File*

1. If at 2. in *Create Local Task* and the system was unable to append the task to the local file in memory.
2. The creator is prompted by the system was unable to append the task.
3. The creator is transitioned back to the main application screen by the system.

**UC02 - View Local Tasks**

**Actors:** Contributor (primary)

**Goal:** After a contributor completes *View Local Tasks*, all local tasks will have been displayed onscreen.

**Trigger:** Contributor has selected the "View Tasks" button from the main screen.

**Pre-conditions:**

* There exists local tasks in memory or on SD card.

**Basic Flow:**

*View Local Tasks*

1. The system loads local tasks from memory.
2. The system updates the progress bars for each task using the tasks current number of task items and expected number of task items.
3. The contributor is displayed the local tasks and progress bars by the system.

**Variations:**

*No Local Tasks in Memory*

1. If at 1. in *View Local Tasks* and there exists no tasks in local memory.
2. The system loads local tasks from the SD card to memory.
3. The use case resumes back at 1. in *View Local Tasks.*

*No Local Tasks on SD Card*

1. If at 2. in *No Local Tasks in Memory* and there exists no local task file or an empty local task file on the SD card.
2. The contributor is prompted by the system that there are no local tasks.

**Post-conditions:**

* The contributor has successfully been displayed all local tasks.

**UC03 - View Local Task**

**Actors:** Creator (primary)

**Goal:** After a creator completes *View Local Task*, the creator has been displayed the information for the task by the system.

**Trigger:** Creator has short pressed a task from the View Local Tasks screen.

**Pre-conditions:**

* The local task exists

**Basic Flow:**

*View and Modify Local Tasks*

1. The creator is displayed local task’s title, creator name, description, type, and number of required items (these fields are modifiable by the creator).
2. As long as the creator modifies a task’s entries, repeat these steps:
   1. The creator inputs a new data value for the field
   2. The system displays the new data value
3. The creator clicks “Save” to indicate they wish to save their modifications to the system.
4. The system overwrites the local task in the list of local tasks memory with the newly modified task.
5. The system overwrites the local task file on the SD card.
6. The creator is prompted by the system that the task was saved successfully.
7. The use case resumes back at 1. in *Create New Task.*

**Variations:**

*Share Task*

1. If before 2. In *View Local Task* and the creator has clicked the share button to indicate they wish the task to be shareable.
2. The system uploads the new task to CrowdSourcer.
3. The system saves the newly created task into the list of shared tasks.
4. The system removes the task from the list of local tasks.
5. The system overwrites the local task file on the SD card.
6. The creator is prompted that the save was successfully and that the task was shared by the system.
7. The creator is transitioned back to the main screen by the system.

**Post-conditions:**

* The contributor has successfully been displayed all local tasks.

**Exceptions:**

*Missing Title or Creator Name*

1. If at 3. in *View and Modify Local Task* or 1. In *Share Task* and the creator has not filled in either the task's title or their name.
2. The creator is prompted by the system which information is missing.
3. The use case resumes back at 1. in *View and Modify Local Task.*

*Incorrect Number of Task Items*

1. If at 3. in *View and Modify Local Task* or 1. In *Share Task* and the creator has not entered an integer that is greater than or equal to 1 for the number of task items.
2. The creator is prompted by the system that the creator has entered an incorrect number of task items.
3. The use case resumes back at 1. in *View and Modify Local Task.*

*Cannot Upload to CrowdSourcer*

1. If at 2. in *Share Task* and the system was unable to upload the new task to CrowdSourcer
2. The creator is prompted by the system was unable to upload the task to CrowdSourcer and that the task is being saved locally instead.
3. The system changes the task to NOT shared.
4. The use case jumps to 4. in *View and Modify Local Task.*

*Cannot Overwrite Local Task File*

1. If at 5. in *View and Modify Local Task* and the system was unable to overwrite the local task file on the SD card.
2. The system restores the task to its previous state in the list of local tasks.
3. The creator is prompted by the system was overwrite local tasks.
4. The creator is transitioned back to the main application screen by the system.

**UC4 - View Shared Tasks**

**Actors:** Contributor (primary), CrowdSourcer

**Goal:** After a contributor completes *View Shared Tasks*, shared tasks created by the contributor or all shared tasks will have been displayed onscreen.

**Trigger:** Contributor has selected the "View Shared Tasks" button from the main screen.

**Pre-conditions:**

* There exists shared tasks in memory or on CrowdSourcer.

**Basic Flow:**

*View Shared Tasks*

1. The system loads shared tasks from memory.
2. The system filters out the tasks that are completed.
3. The system filters the tasks to be only those tasks whose CreatorId matches that of the contributors Device ID.
4. The system updates the progress bars for each task using the tasks current number of task items and expected number of task items.
5. The contributor is displayed the existing tasks and progress bars by the system.

**Variations:**

*Task completed*

1. If at 3. In *View Shared Tasks* and there are

*No Shared Tasks in Memory*

1. If at 1. in *View Shared Tasks* and there exists no shared tasks in local memory.
2. The system calls the use case **Synchronize Tasks**.
3. The use case resumes back at 1. in *View Shared Tasks.*

*No Personal Shared Tasks*

1. If at 3. in *View Shared Tasks* and there exists no shared tasks created by the contributor’s device.
2. The contributor is prompted by the system that there are no shared tasks created by their device.
3. The use case resumes back at 3. In *View Shared Tasks.*

*No Shared Tasks in CrowdSourcer*

1. If at 2. in *No Shared Tasks in Memory* and there exists no shared tasks in CrowdSourcer
2. The contributor is prompted by the system that there are no shared tasks.
3. The use case resumes back at 4. In *View Shared Tasks.*

*View My Tasks*

1. If at any point in *View Shared Tasks* or *View All Tasks* the contributor changes the toggle to “My Tasks”
2. The system applies a Device ID filter to the displayed tasks (if filter doesn’t exist).
3. The use case resumes back at 3. In *View Shared Tasks.*

*View All Tasks*

1. If at any point in *View Shared Tasks* or *View My Tasks* the contributor switches the toggle to “All Tasks”.
2. The system removes the Device ID filter from displayed tasks (if filter exists).
3. The use case resumes back at 3. In *View Shared Tasks*.

**Post-conditions:**

* The contributor has successfully been displayed all local tasks.

**Exceptions:**

*Unable*

**UC5 - View Shared Task**

**Actors:** Contributor (primary)

**Goal:** After a contributor completes *View Shared Task*, the contributor has been displayed the information for the task by the system.

**Trigger:** Contributor has short pressed a task from the View Local Tasks screen.

**Pre-conditions:**

* The local task exists

**Basic Flow:**

*View and Modify Local Tasks*

1. The creator is displayed local task’s title, creator name, description, type, and number of required items (these fields are modifiable by the creator).
2. As long as the creator modifies a task’s entries, repeat these steps:
   1. The creator inputs a new data value for the field
   2. The system displays the new data value
3. The creator clicks “Save” to indicate they wish to save their modifications to the system.
4. The system overwrites the local task in the list of local tasks memory with the newly modified task.
5. The system overwrites the local task file on the SD card.
6. The creator is prompted by the system that the task was saved successfully.
7. The use case resumes back at 1. in *Create New Task.*

**Variations:**

*Share Task*

1. If at 3. In *View Local Task* and the creator has previously indicated they wish the task to be shareable.
2. The system uploads the new task to CrowdSourcer.
3. The system saves the newly created task into the list of shared tasks.
4. The system removes the task from the list of local tasks.
5. The system overwrites the local task file on the SD card.
6. The creator is prompted that the save was successfully and that the task was shared by the system.
7. The creator is transitioned back to the main screen by the system.

**Post-conditions:**

* The contributor has successfully been displayed all local tasks.

**Exceptions:**

*Missing Title or Creator Name*

1. If at 5. in *Create New Task* and the creator has not filled in either the task's title or their name.
2. The creator is prompted by the system which information is missing.
3. The use case resumes back at 1. in *View and Modify Local Task.*

*Incorrect Number of Task Items*

1. If at 5. in *Create New Task* and the creator has not entered an integer that is greater than or equal to 1 for the number of task items.
2. The creator is prompted by the system that the creator has entered an incorrect number of task items.
3. The use case resumes back at 1. in *View and Modify Local Task.*

*Cannot Upload to CrowdSourcer*

1. If at 2. in *Share Task* and the system was unable to upload the new task to CrowdSourcer
2. The creator is prompted by the system was unable to upload the task to CrowdSourcer and that the task is being saved locally instead.
3. The system changes the task to NOT shared.
4. The use case jumps to 4. in *View and Modify Local Task.*

*Cannot Append Task to Local File*

1. If at 2. in *View and Modify Local Task* and the system was unable to append the task to the local file in memory.
2. The creator is prompted by the system was unable to append the task.
3. The creator is transitioned back to the main application screen by the system.

**UC6 - Add Text Task Item**

**Actors:** Contributor (primary)

**Goal:** After a contributor completes *Add Text Task Item*, the contributor has successfully added a task item to the previously selected task.

**Trigger:** Contributor has selected “Add Task Item” in any of the view task screens.

**Pre-conditions:**

* The contributor has selected a task.
* The task exists.

**Basic Flow:**

*Add Text Task Item*

1. The contributor is displayed a pop up dialog by the system.
2. The contributor enters the text for the task item into the system.
3. The contributor clicks “Save” to indicate to the system they wish for the text task item to be added.
4. The system adds the new text task item to the chosen task in the list of local tasks in memory.
5. The system overwrites the local task file on the SD card.
6. The creator is prompted by the system that the task was saved successfully.
7. The system destroys the dialog and refreshes the task list.

**Variations:**

*Add to Shared Task*

1. If at 3. In *Add Text Task Item* and the chosen task is a shared task.
2. The system adds the new text task item to the chosen task in the list of shared tasks in memory.
3. The system updates the chosen task in CrowdSourcer.
4. The use case resumes at 6. In *Add Task Item*.

**Post-conditions:**

* The contributor has successfully been displayed all local tasks.

**Exceptions:**

*Cannot Update Task in CrowdSourcer*

1. If at 3. in *Add to Shared Task* and the system was unable to update the task in CrowdSourcer
2. The contributor is prompted by the system that the system was unable to update the task in CrowdSourcer.
3. The system removes the text task item from the task.
4. The use case resumes at 7. In *Add Task Item*.

*Cannot Overwrite Local File*

1. If at 5. in *Add Text Task Item* and the system was unable to overwrite the local task file on the SD card.
2. The creator is prompted by the system was unable to overwrite the file.
3. The system removes the text task item from the task.
4. The use case resumes at 7. In *Add Task Item*.

**UC7 - Add Audio Task Item**

**Actors:** Contributor (primary)

**Goal:** After a contributor completes *Add Audio Task Item*, the contributor has successfully added a task item to the previously selected task.

**Trigger:** Contributor has selected “Add Task Item” in any of the view task screens.

**Pre-conditions:**

* The contributor has selected a task.
* The task exists.

**Basic Flow:**

*Add Audio Task Item*

1. The contributor selects “Record” to indicate to the system to begin recording a new audio clip.
2. The system begins recording a new audio clip.
3. The contributor selects “Stop” to indicate the system to stop recording.
4. The system stops recording the audio clip.
5. The contributor clicks “Save” to indicate to the system they wish for the text task item to be added.
6. The system adds the new text task item to the chosen task in the list of local tasks in memory.
7. The system overwrites the local task file on the SD card.
8. The creator is prompted by the system that the task was saved successfully.
9. The system destroys the dialog and refreshes the task list.

**Variations:**

*Add to Shared Task*

1. If at 3. In *Add Text Task Item* and the chosen task is a shared task.
2. The system adds the new text task item to the chosen task in the list of shared tasks in memory.
3. The system updates the chosen task in CrowdSourcer.
4. The use case resumes at 6. In *Add Task Item*.

**Post-conditions:**

* The contributor has successfully been displayed all local tasks.

**Exceptions:**

*Cannot Update Task in CrowdSourcer*

1. If at 3. in *Add to Shared Task* and the system was unable to update the task in CrowdSourcer
2. The contributor is prompted by the system that the system was unable to update the task in CrowdSourcer.
3. The system removes the text task item from the task.
4. The use case resumes at 7. In *Add Task Item*.

*Cannot Overwrite Local File*

1. If at 5. in *Add Text Task Item* and the system was unable to overwrite the local task file on the SD card.
2. The creator is prompted by the system was unable to overwrite the file.
3. The system removes the text task item from the task.
4. The use case resumes at 7. In *Add Task Item*.

**UC8 - Add Image Task Item**

**Actors:** Contributor (primary)

**Goal:** After a contributor completes *Add Image Task Item*, the contributor has successfully added a task item to the previously selected task.

**Trigger:** Contributor has selected “Add Task Item” in any of the view task screens.

**Pre-conditions:**

* The contributor has selected a task.
* The task exists.

**Basic Flow:**

*Add Audio Task Item*

1. The contributor selects “Take Picture” to indicate to the system to start the camera application.
2. The system starts the camera application.
3. The contributor touches the screen to indicate to the system to take a picture
4. The system takes a picture.
5. The system displays the taken picture on screen.
6. The contributor clicks “Save” to indicate to the system they wish for the text task item to be added.
7. The system adds the new text task item to the chosen task in the list of local tasks in memory.
8. The system overwrites the local task file on the SD card.
9. The creator is prompted by the system that the task was saved successfully.
10. The system destroys the dialog and refreshes the task list.

**Variations:**

*Add to Shared Task*

1. If at 3. In *Add Text Task Item* and the chosen task is a shared task.
2. The system adds the new text task item to the chosen task in the list of shared tasks in memory.
3. The system updates the chosen task in CrowdSourcer.
4. The use case resumes at 6. In *Add Task Item*.

**Post-conditions:**

* The contributor has successfully been displayed all local tasks.

**Exceptions:**

*Cannot Update Task in CrowdSourcer*

1. If at 3. in *Add to Shared Task* and the system was unable to update the task in CrowdSourcer
2. The contributor is prompted by the system that the system was unable to update the task in CrowdSourcer.
3. The system removes the text task item from the task.
4. The use case resumes at 7. In *Add Task Item*.

*Cannot Overwrite Local File*

1. If at 5. in *Add Text Task Item* and the system was unable to overwrite the local task file on the SD card.
2. The creator is prompted by the system was unable to overwrite the file.
3. The system removes the text task item from the task.
4. The use case resumes at 7. In *Add Task Item*.

**UC9 - View Completed Tasks**

**Actors:** Contributor (primary)

**Goal:** After contributor completes *View Completed Tasks,* all completed tasks will have been displayed on-screen.

**Triggers:**

* Contributor has selected the “View Completed Tasks” button from the main screen.

**Pre-conditions:**

* There exists completed tasks in memory, the SD card or on CrowdSourcer.

**Basic Flow:**

*View Completed Tasks*

1. The system loads completed tasks from memory.
2. The contributor is displayed the completed tasks by the system.

**Variations:**

*No Completed Tasks in Memory*

1. If at 1. In *View Completed Tasks* and there exists no completed tasks in memory.
2. The system loads completed task from the task file on the SD card to the completed task list in memory.
3. The system loads completed task from CrowdSourcer to completed task list in memory.
4. The use case resumes back at 1. In *View Completed Tasks.*

**Post-conditions:**

* The contributor has successfully been displayed all completed tasks

**UC10 – View Completed Task**

**Actors:** Creator (primary)

**Goal:** After a creator completes *Notify Task Completed* the will have been transitioned to view the completed task they were notified about from the system.

**Triggers:**

* The creator has clicked on a task completed push notification sent by the system.
* The creator has short pressed on a completed task in the completed tasks list.

**Pre-conditions:**

* A task has been completed.
* The system has sent a push notification to the creator that a task has been completed.

**Basic Flow:**

*Complete Task*

1. The system dismisses any existing task completed push notifications.
2. The system displays the completed task’s title, description, creator name, and number of required task items to the creator.
3. The system displays the task’s items in a list to the creator.

**Variations:**

*Reject Task*

1. If at. Any point during *Complete Task* the creator selects the “Reject” button to indicate to the system that they reject the completed task.
2. The system creates a new task in the local tasks with the same title, description, creator name and number of required items as the rejected task.
3. The system removes the rejected task from the completed task list.
4. The system overwrites the SD card file with the modified tasks.
5. The creator is notified by the system that the task has been rejected.
6. The creator is transitioned to the completed task list by the user.

**Post-conditions:**

* The system has successfully downloaded all tasks from CrowdSourcer

**UC11 - Synchronize Tasks**

**Actors:** Contributor (primary), CrowdSourcer

**Goal:** After a contributor completes *Synchronize Tasks* the system will have reloaded the shared tasks from CrowdSourcer.

**Trigger:** Creator has selected the Synchronize button at the top right of the screen.

**Pre-conditions:**

* The system is able to access CrowdSourcer.

**Basic Flow:**

*Synchronize Tasks*

1. The system downloads shared tasks from CrowdSourcer.
2. The system replaces existing shared tasks in memory with those downloaded from CrowdSourcer.
3. The contributor is prompted by the system that the tasks were synchronized successfully.

**Variations:**

Post-conditions:

* The system has successfully downloaded all tasks from CrowdSourcer

**Exceptions:**

*Cannot Download from CrowdSourcer*

1. If at 1. In *Synchronize Tasks* and the system was unable to download tasks from CrowdSourcer.
2. The contributor is prompted by the system that the system was unable to reach CrowdSourcer.

**UC12 – Task Completed Notification**

**Actors:** Creator (primary)

**Goal:** After a creator completes *Notify Task Completed* the will have been transitioned to view the completed task they were notified about from the system.

**Trigger:** The creator has clicked on a task completed push notification sent by the system.

**Pre-conditions:**

* A task has been completed.
* The system has sent a push notification to the creator that a task has been completed.

**Basic Flow:**

*Send to Completed Task*

1. The creator is
2. The contributor is prompted by the system that the tasks were synchronized successfully.

**Variations:**

Post-conditions:

* The system has successfully downloaded all tasks from CrowdSourcer

**Exceptions:**

*Cannot Download from CrowdSourcer*

1. If at 1. In *Synchronize Tasks* and the system was unable to download tasks from CrowdSourcer.
2. The contributor is prompted by the system that the system was unable to reach CrowdSourcer.