U.Plan() Meeting Minutes 5/10/17

# Timing: 9:40 – 11:00

|  |  |
| --- | --- |
| **Number** | **Details** |
| 1 | Discussion on Purpose of U.Plan()   * We want to improve the lives of future generations. * Specifically working population. * **Make sure that nothing on users’ calendar clashes** * **And that end users can keep track of their work.** * Our project would be **Android-Only**. |
| 2 | Discussion on Features of U.Plan()   * There would be a calendar interface as the MainActivity * MainActivity.java would implement a ListView for its calendar interface, which would show what times the end user has things on * Note that the ListView would only show, what the user has for 1 day. This comprises both tasks and events * The AppBar for MainActivity would have a **search** icon * The search icon would trigger a “search for…” feature. * The “search for…” feature will search for events between a time frame (e.g. from 1 May to 4 May), so that if let’s say the user’s boss wants to place more commitments onto the user, the user can just search very quickly and determine firsthand if he has clashing commitments * The “search for…” feature will also search for **NAME**. |
| 3 | Discussion on Spinner   * Since the ListView only displays what the user has in 1 day, * There will be a spinner consisting the days in the months near the left hand side * The spinner can be triggered by swiping from the far left edge of the screen towards the right * This spinner is to be placed in the Navigation Bar Activity * We could use com.aigestudios.dayPicker to implement this Spinner. * When a new day is selected the content for that day will be loaded * The side bar will have the spinner below and a gear icon on top for the settings. |
| 4 | Discussion on Switching between Months   * Since the spinner is activated by swiping from the far left, * In order to switch between months, the screen needs to be swiped from the middle onwards. * This behavior would be similar to the behavior of TabLayout. * Hence, the name of the AppBar would be the month. |
| 5 | Discussion on Adding Events / Task   * There would be a **FloatingActionButton** to add events / tasks * Both of the events / tasks will also have these fields:   + Name   + Time Duration (XXX-XXX)   + Type (this will not be included in the database, but it determines which type of ‘date’ allocation you get   + Date * For Events, since they are rigid, then the user will be prompted to input a rigid date and rigid time * For Tasks which can be dynamically allocated, the program will search for the day with the least things and suggest for you, if you don’t like you can manually use it. |
| 6 | Discussion on Notifications   * We might use AlarmManager for notifications * Or queue the time a notification will arrive with the Notification class * Zerui to check if this is possible |

Database Structure

1. Database (:root)
   1. Day
      1. Unique Identifier for the Day (\_id auto increment, start from 0)
      2. Task\_Count (for the allocation)
   2. Tasks/Events
      1. Day field (use this for the “search” function and also to associate which tasks u should load for which day. The AlarmManager will also use this to send notification)
      2. Task/Event Name field
      3. Task/Event Time field