EOE Kanban – High Level Design

Prepared by : Erez Shmuely, Or Kandabi, Elad Solomon

1. Introduction

This document describes the high level design for the EOE Kanban project. The EOE Kanban allow you to focus your priorities, keep your team in the loop about what's current and what's coming up. A Kanban contains a board that allows organizing your cards by columns of "to-do", "in progress" and "done" column categories. The Kanban board contains Cards, where each card resides in a column. The Kanban intended for big and small companies. Everyone's needs are unique, but the upside of the Kanban method is that, once you apply it, you can standardize and optimize your workflow with a relative ease.

2. Deliverables

The clients of this software are big and small companies (also individuals) that want to control the workflow of their projects, minimize multitasking, streamline their work in progress efficiency, and improve the speed and quality of the work their collaborative and self-managing team produces.

The users that have access to the software are all the people that taking part at a projects, and include workers from the most junior to the highest ranked manger in a company.

3. Main goal

The primary objective of the project is to control a steady workflow, manage agile development and helps you to visualize your workflow.

4. Dictionary

user[1]- every registered member that have access to the project.

manager[2]- CEO ,general manager and team head.

priority[3]-the priority will determinate by the manager of the project according to the needs by the moment.

malformed input[4]- input that disobey the guidelines orders (e.g. above 300 chars in a card, unavailable date, adding to much tasks for a limited column, etc.).

5. Requirements

5.1 Functional

4.1.1. Users Login

- A user password must be in length of 4 to 20 characters and must include at least one capital character, one small character and a number.
- Each email address must be unique in the system.
- The program will allow registering of new users.
- The program will allow login and logout of existing users.
- The login will succeed if, and only if, the email and the password matches the user's email and password.

4.1.2. Boards

- A user can add and remove boards.
- By default, a board has three columns: 'backlog', 'in progress' and 'done'.
- Columns can be added to the board and be removed.
- Columns presentation order can be changed.
- Each column should support limiting the maximum number of its tasks.
- By default, there will be no limit on the number of the tasks.
- The board will support adding new tasks to its leftmost column only.
- Tasks can be moved from one column to the next column in the presentation order. No other movements are allowed.

4.1.3. Tasks

- A task can be changed by the user, unless it arrived to the final column in the representation order.
- All of the task data can be changed, except for the creation time.
- Tasks can be sorted by date.
- Tasks can be filtered by its text fields (i.e. the title and/or the description). For example: filtering the tasks by the search term "login" will return tasks that their title/description include the word login, as:

  a. Implement login screen.

b. To add captcha to the login screen.

5.2 Non-Functional

4.2.1. Persistency:

- The DAL (persistence layer) should use a DBMS (SQLite)
- The following data should be persistent:
  1. The users
  2. The boards
  3. The tasks
  4. The columns
- Persistent data should be restored once the program starts.

4.2.2. Logging:

You must maintain a log which will track all errors in the system. Note that "error" does not necessarily mean an exception (see next item) that is "thrown" or "raised" by your runtime environment, but any situation which counts as invalid in our domain. Some guidelines:

- Tag entries with their severity / priority
- Provide enough information to understand what went wrong, and where
- Avoid storing entire stack traces directly in the log (they are more verbose than useful)
- Use a shelf Logger. Some examples: log4net, Observer logger, Composition Logger, end there are many more.

4.2.3. Exception Handling is the process of responding to the occurrence, during computation, of exceptions – anomalous or exceptional conditions requiring special processing – often changing the normal flow of program execution. the program will operate even when error occurs:

- Handle any malformed input.
- Handle logic errors (e.g., login for non-existing user, etc.).

4.2.4. UX

- Buttons in the interface should contain only short text descriptions (no more than two words) or a descriptive icon (preferably both).

- Text fields and text boxes should be large enough to comfortably accommodate user input, but not so large that they are unwieldy.
- Data input by users should be validated, and input of invalid characters should be blocked (e.g. empty messages, nicknames, etc.).

4.2.5. Tests: the program will be checked by the Unit Test Framework.