Test Project
Session 3

IT SOFTWARE SOLUTIONS FOR BUSINESS

Submitted by:
Independent Test Project Design Team

WSC2019_TP09_S3_EN

Contents

This Test Project proposal consists of the following documentation/files:

1. WSC2019_TP09_S3_EN.pdf	(Session 3 instructions)
2. Session3-MySQL.sql	(SQL Script to create tables with data for MySQL)
3. Session3-MsSQL.sql	(SQL Script to create tables with data for Microsoft SQL)

Introduction

Due to the large-scale expansion at Kazan Neft, the company has decided to develop an in-house maintenance management and enterprise asset management system. As part of such a system, you are required to develop mobile device applications to handle preventive maintenance tasks.

The assets are subjected to a regular schedule of maintenance tasks, such as inspections, cleaning, lubrication, adjustments and calibration where are hereby called preventive maintenance tasks.

Listed below are some of the key attributes of preventive maintenance tasks:

- *Routine* - The work is performed on a routine basis regardless of whether functionality or performance of the asset is degraded.
- *Consistent* - The frequency of the maintenance is generally constant and is usually based on the expected life of the components being maintained.
- *Fixed Intervals* - The maintenance is carried out at predefined intervals in an attempt to reduce equipment failures or to ensure a consistent appearance of the assets.

Description of Project and Tasks

While developing the test project, please make sure the deliverables conform to the basic guidelines drawn out by different departments at Kazan Neft:

- There should be consistency in using the provided style guide throughout development.
- All required software modules must have applicable and useful validation and error messages as expected by the industry.
- Offer a scrollbar if the number of records on a list or a table that do not fit in the form area comfortably. Hide scrollbars if all content can comfortably be displayed.
- The de-facto standard, ISO compliant date format is YYYY-MM-DD which will be used in this task where applicable.
- Where applicable, use comments in code to have the code more programmer-readable.
- The use of valid and proper naming conventions is expected in all material submitted.
- When a form or a dialogue is in focus, operations on other forms need to be suspended.
- The caption of Delete and Cancel buttons need to be in red to help with accidental mishaps.
- When using colours to differentiate between rows or records, there needs to be visible clarification on the screen as to what they stand for.
- The wireframe diagrams provided as part of this document are only suggestions and the solution produced does not have to be, in any way, mirror what has been pictured.
- Time management is critical to the success of any project and so it is expected of all deliverables to be complete and operational upon delivery.
- The user interface of the current task needs to be implemented on the Android platform and will only be accepted on the mobile devices provided.

- As an industry standard, the company infrastructure is based on a central database and the application should be designed to process all their data requests through a Web API. The company will provide all necessary specifications for you to model and deploy the data interface.

Instructions to the Competitor

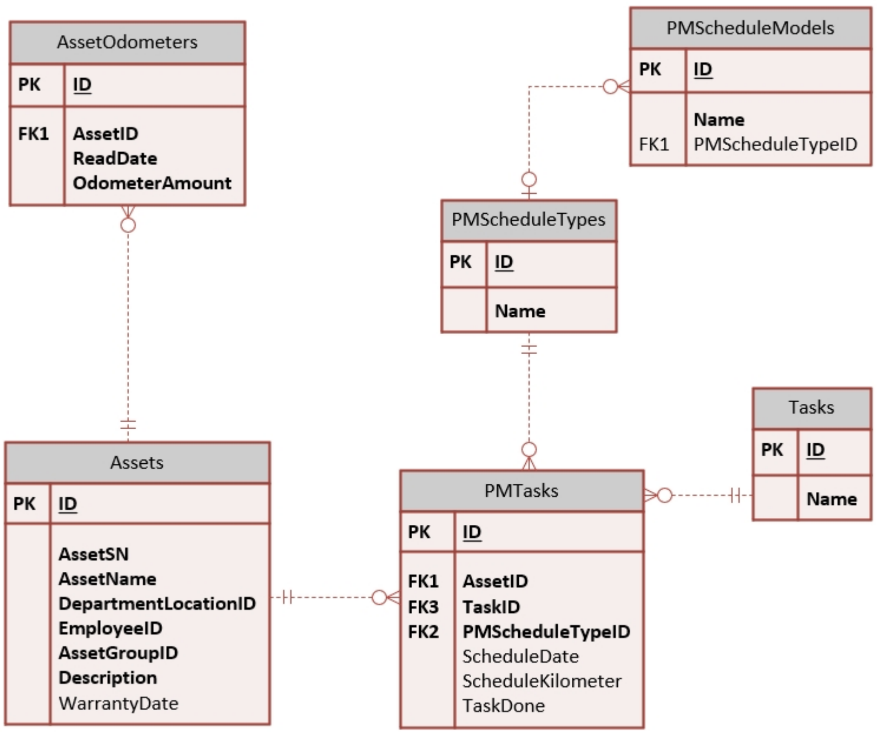
3.1 Connecting to the Database

Use a database by the name of “Session3” in your desired RDBMS Platform (MySQL or Microsoft SQL Server). This will be the main and only database you will use in this session.

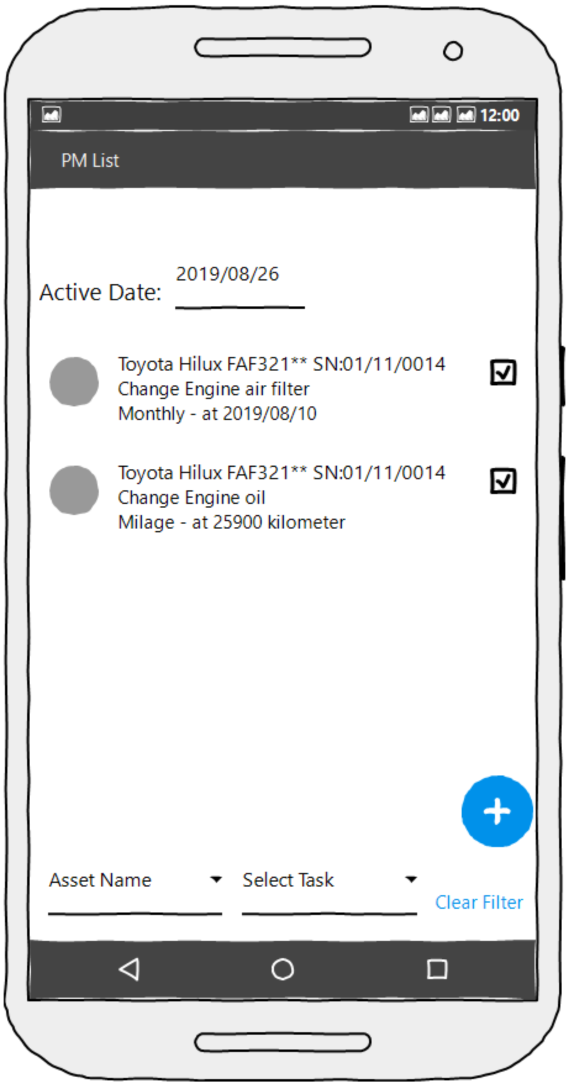
3.2 Importing Database Structure

Depending on your preferred RDBMS platform, a SQL scripts is made available. The said scripts consist of the database structure and data required to complete the tasks. The data needs to be imported to the database created for this session named “Session3”.

As instructed by the designers, the database structure provided for the purpose of this section cannot be altered. This applies to removal of tables, adding or deleting any fields on the tables or of change in their data types.



To help further perceive the thinking behind the structure of the database, the database designers provide an Entity-Relationship Diagram (ERD). The aforementioned diagram explains the conceptual and representational model of data used in the database.


12:00

PM List

Active Date: 2019/08/26

Toyota Hilux FAF321** SN:01/11/0014
Change Engine air filter
Monthly - at 2019/08/10

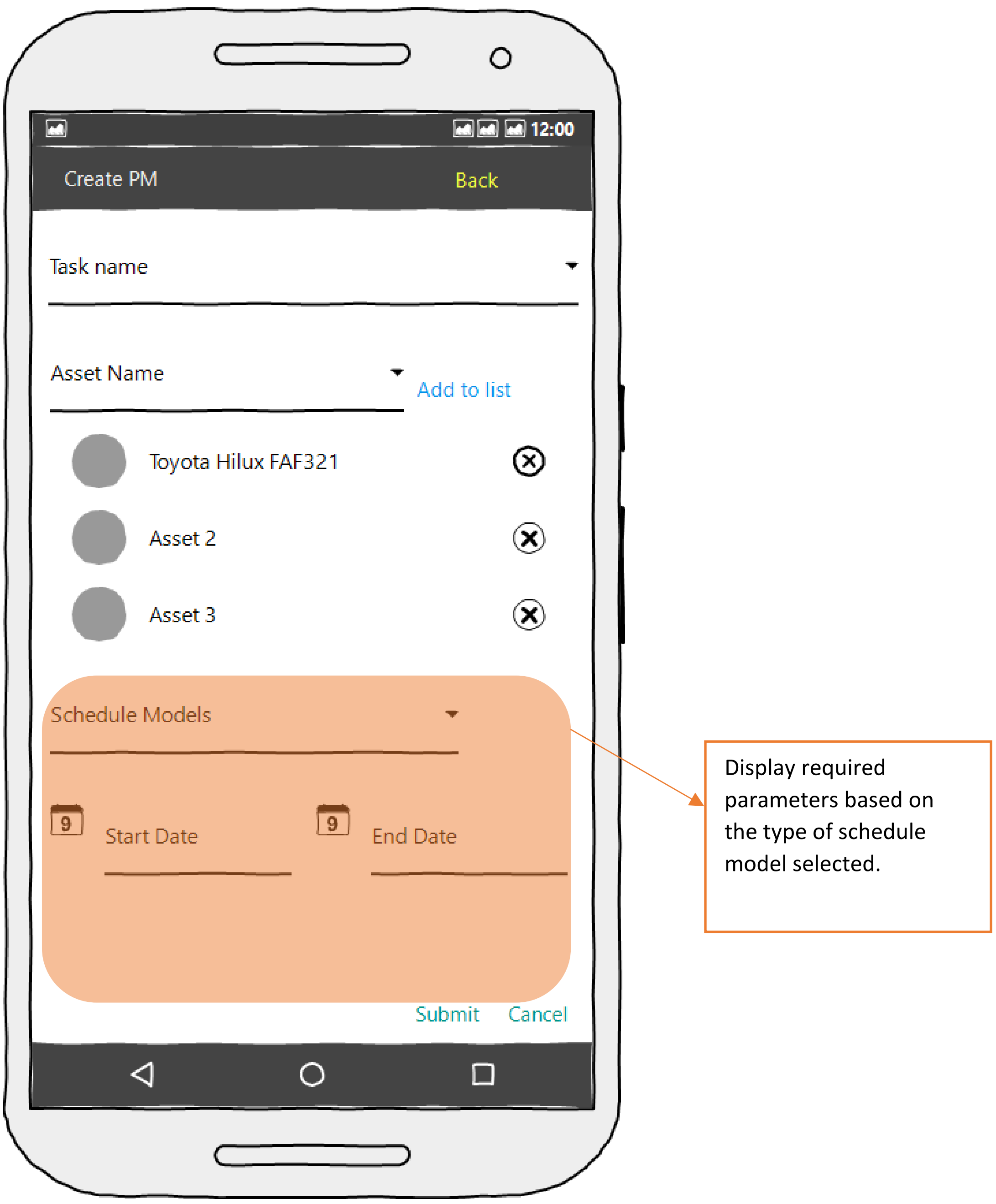
Toyota Hilux FAF321** SN:01/11/0014
Change Engine oil
Milage - at 25900 kilometer

3.3 Active Preventive Tasks

This would be the first page the user sees and can use it to access functionalities of other section of the application. This form will work as follows:

- The following data will be displayed on the form:
    - Asset Name, Asset SN, Task Name, Schedule Type, (ScheduleDate or ScheduleKilometer)
- The top of the form an entry is place that goes by the name of “Active Date”.
    - By default, the field should be set to the current system date.
    - By changing the date, the active tasks list should be refreshed to reflect the changes.
- In the middle, a scrollable list of all the active tasks are displayed:
    - There are two kinds of active tasks:
        - Time-based active tasks are tasks which are based on a time schedule (ScheduleDate). They schedule will be compared to the selected date. There are three categories for these kinds of tasks:
            - The tasks that are past due (scheduled date is before active date).
            - The tasks that are due on the date of active date.
            - Task that are scheduled for the next four days.

        - Run-based active tasks are those that are due because the threshold or the condition for the task has been met. For the purpose of this project we will check against vehicle milage (ScheduleKilometer).
    - A checkbox next to each task indicates whether the task has been attended to or processed:
        - By clicking on the checkbox, the checkmark toggles between empty checkbox (not done or unprocessed) and checkbox with a checkmark (processed or done).
        - The data for the checkbox is stored and retrieved from the field "TaskDone" where true means that the task has been processed.
        - Upon any change, the active tasks list should get an instant update to reflect it.
    - The list is organized in the following order:
        - The run-based active tasks that are not processed come first.
        - Time-based active tasks that are past due according to the active date on the form and not processed are listed next.
        - The list continues with unprocessed tasks due on the actual active date.
        - The time-based tasks that are due in the four days following the active date and are yet to be processed will be listed as the next priority.
        - The processed (ones with checkmark) should be listed at the very end.
    - The formatting provided to distinguish between different records are as follows:
        - The color used for run-based active tasks are black if unprocessed and grey if processed.
        - The formatting for the time-based active tasks are as follows:
            - The user will see records that are past due in orange if processed and red otherwise.
            - The color for the ones on the active date and unprocessed are black and if processed they would be green.
            - The ones that are on the four-day period after active date are purple if unprocessed, otherwise the color black will be used.
- The user will be able to filter out preventive tasks using two filters at the bottom of the form:
    - Any change to the field will immediately refresh the active tasks list to reflect them.
    - The asset name drop-down list is populated from the database:
    - The select task drop-down list is populated from the database.
    - A button labelled "Clear filter" removes all filters from the active tasks list and reset the two drop-down boxes to their defaults.
- A floating add new task button (⊕) at the bottom right corner of the screen helps the user add new preventive maintenance tasks as described in the next section.



3.4 Registering New Preventive Maintenance Tasks

By using the add new preventive maintenance task floating button (+) on the main form the user will have the option to define and add new preventive maintenance tasks as explained here:

- The following information are used in this section:
  - Asset Name, Task Name, Task Start Date, Task End Date, Schedule Model, Schedule Parameter(s)
- The schedule models drop-down is populated from the database with one of the listed models below.
  - Time-based assets are recurring tasks with defined intervals in the following three categories:
    - Daily intervals model refers to recurring tasks that are described by submitting the gap in number of days they should run. For example, you can have a task that need to be completed every twelve days.

        - Weekly intervals model refers to recurring tasks that need to be described by defining the day of the week the tasks needs to take place and the number of weeks between each recurrence. For example, you can have a task that needs to be repeated every three weeks on a Monday.
        - Monthly intervals model refers to recurring tasks are defined by entering the day of month and the gap as in the form of the number of months. For example, you can have a task that runs every other month on the third day of the month.
    - Run-based tasks are tasks that depend on different characteristics of the assets. For the purpose of the application requested, the parameters evaluated are only the readings on the odometer and how many kilometers a vehicle runs before the task needs to be performed. For example, a certain vehicle might need an oil change every 5000 kms on their clock from the odometer reading of 20000 to 50000. The following is a summary of how it works:
        - The accountable party for the vehicle reads the number of kilometers on the odometer on set dates and they will go on the “AssetOdometers” table. For example, on the 10th the reading shows 1000 kms on the odometer and on the 15th the number on the database shows 1200 that means the distance travelled in the five days has been 200 kms.
        - A start and end range for the recurring task needs to be defined for it to be
        - The user cannot add a task on a range more than once. For example, if there is an oil change set up for every 5000 kms from the readings of 20000 to 50000, the user should not be able to add another recurring oil change between the readings of 30000 and 50000.
- The start and end dates are only valid on the time-based schedule models and define the time range they can work within. For example, if a weekly interval schedule is defined with start date and end dates of seven days, it will only run once but if the start and end dates are more than 21 days, it would run a minimum of three times.
- The “Back” button at the top and the “Submit” and “Cancel” button at the bottom will get the user back to the main menu. The “Submit” button stores the changes to the database and refreshes the main form to reflect the changes while the other two do not.