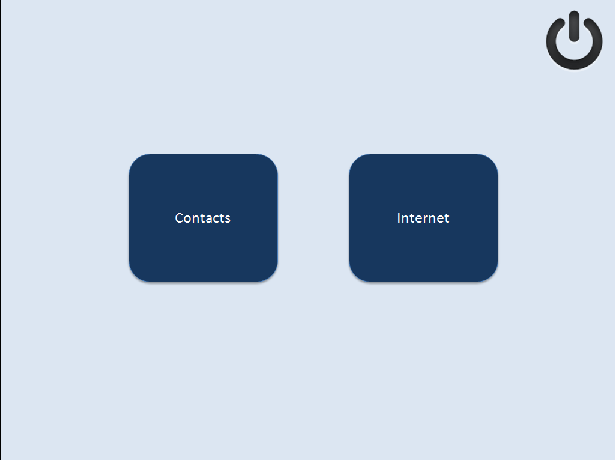
# QT-QML Assignment

**Duration:** 8 hours

This aim of this assignment is to develop a functional QT application, using QML for development of the User Interface, QT for the business logic and QDBus for IPC.

## Requirements in detail

## Startup Screen



*Fig 1: Startup Screen*

The startup screen of the application shall look like in fig 1. It is to follow this spec:

- Fit onto a screen size of 640 x 480

- Colors for various widgets as given in the diagram

- The Button has two states (state shown in fig 1 is normal)

a) Normal : Blue Color (as in fig 1)

b) Clicked: Light Blue (RGB(150,150,255))

- There shall be an animation for the transition of the colors when state changes, duration of 200ms.

Any other animation that you would like to apply for button state transition, is encouraged. Poweroff on right top shall exit the application.

When the Contacts button is pressed, the app has to navigate to the Contact List Screen (fig2).

## Contact List Screen

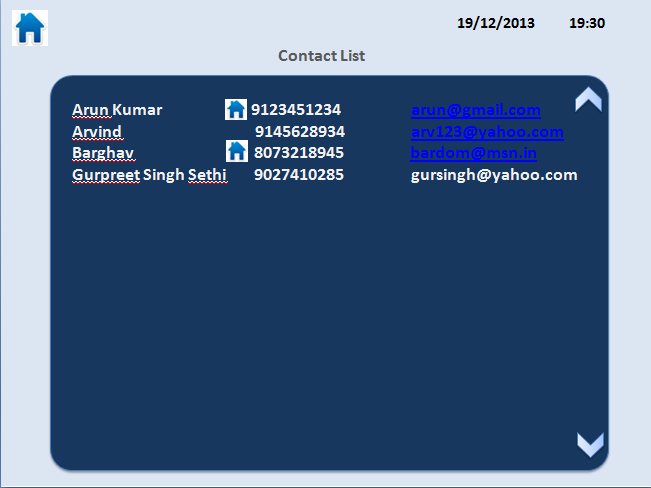


fig2: Contact List

When the Contact button (in fig 1) is clicked a signal has to be emitted (in QML) which has to be handled in a QT class (lets call it as UIController). This class shall further pass this to a QAppManager class. More on this call in section on (***Getting Contact List***). The UIController is responsible for the transition between screens.

The contacts shall be populated in a Model Class (C++) and bound to a listView in QML. The 3 information to be displayed would be fname-lname, mobile, email. (irrespective of full info obtained ). The UI coloring and other aesthetics has to be maintained as given in fig:2 (except for the blue color for a few email ids, use white for them).

The date and time on the right top has to be updated from C++ file and has to be updated every minute!

You may use setContextProperty() and either pull from QML or push from QT Class. You will need to use signals to communicate between QML and QT class.

The Home button on left top takes user back to first (fig:1) screen.

## Getting Contact List

When the Contact button is clicked (fig1), the UIController shall route this call to a QAppManager class. Here, this client application shall notify via QDBus the request for a contact list to a server application.

A QDBus server app should be developed, shall send a QList<ContactInfo> of contact information which it shall obtain from a XML file.

sample XML data:

<contactbook>

<user>

<fname>Arvind</fname>

<mname>Raghav<mname>

<lname>Chari</lname>

<mob>9890123456</mob>

<email>arv@gmail.com</email>

</user>

<user>

.

.</user>

This info shall be read by the DBus server and returned to the client app (which is our actual app) which requested for contact list.

Once the data is received (which shall be in a ContactList struct format), it shall be put in a model which would be bound to the ListView (fig:2)

## Internet Screen

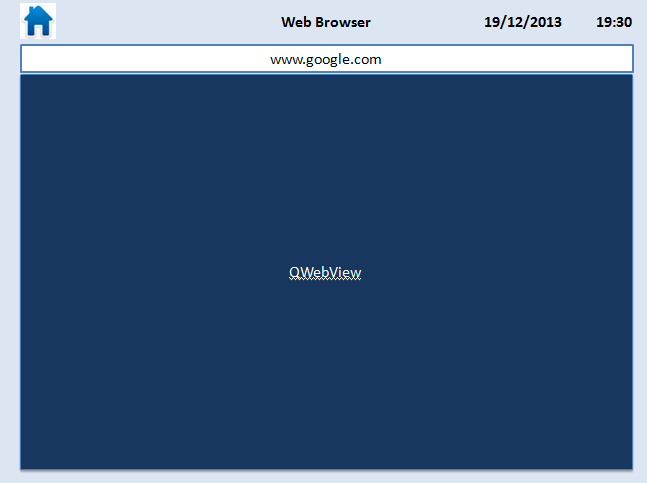


fig 3: Internet Browser

From the home screen when the Internet button is pressed, app shall navigate to this screen. It shall contain a WebView and an addressbar, where user can enter a website and press enter to view it in the webview. Other functionalities remain the same.