****

**PES UNIVERSITY**

***A Project based learning report***

**TOPIC: IMPLEMENTATION OF POINT OF SALE**

**Submitted in Partial fulfilment of the Requirements for VI Semester**

**Bachelor of Technology in Electronics and Communication**

**Jan. - Apr. 2018**

**Under the guidance of**

**Mrs. Prajeesha**

**Asst. prof, Dept of ECE**



**DEPARTMENT OF ELECTRONICS AND COMMUNICATION**

1. **ENGINEERING**

**PES UNIVERSITY, BANGALORE-85**

**TEAM MEMBERS**

**CHIRANJEEVI N – 01FB16EEC083**

**CHINMAY D KUCHINAD – 01FB16EEC081**

**D G SUHAAS KIRAN – 01FB16EEC085**

**JOSEPH SUNIL – 01FB16EEC113**

**POS - Point of sale**

**INTRODUCTION**

Any business selling in person will have a ‘point of sale’, commonly abbreviated ‘POS’.

***POS is the setup you have in place for processing face-to-face payments from customers****.*

POS is a constellation of things that together enable you to process customer-facing transactions efficiently and streamline business processes connected with your sales.

The setup will vary in look and functionality depending

1. On your choice of technology. 2. What payment methods you accept (whether you print paper receipts). 3. How you record sales and organize end-of-day bookkeeping, and the inventory systems you have in place for your products.

**BACKGROUND**

Earlier, a point of sale system was just a cash register. The person operating the till would manually enter the prices of purchased items, often with the help of price tickets. Further POS became more computerized, storing a product database on a computer server. A barcode reader was used to avoid manual price entry and store transaction details electronically. Now advanced cloud-based POS systems are being used where data is stored online.

GRAPHIC USER INTERFACE - wxPython

* The **graphical user interface** (**GUI** )is a form of [user interface](https://en.wikipedia.org/wiki/User_interface) that allows [users](https://en.wikipedia.org/wiki/User_(computing)) to [interact with electronic devices](https://en.wikipedia.org/wiki/Human%E2%80%93computer_interaction) through graphical [icons](https://en.wikipedia.org/wiki/Computer_icon) and visual indicators such as secondary notation, instead of [text-based user interfaces](https://en.wikipedia.org/wiki/Text-based_user_interface), typed command labels or text navigation.
* In this project we are using wxPython for graphical user interface.
* **wxPython** is a [wrapper](https://en.wikipedia.org/wiki/Wrapper_library) for the [cross-platform](https://en.wikipedia.org/wiki/Cross-platform) [GUI](https://en.wikipedia.org/wiki/Graphical_user_interface) [API](https://en.wikipedia.org/wiki/Application_programming_interface)  for the [Python programming language](https://en.wikipedia.org/wiki/Python_(programming_language)). It is implemented as a Python [extension module](https://en.wikipedia.org/w/index.php?title=Python_extension_module&action=edit&redlink=1) ([native code](https://en.wikipedia.org/wiki/Native_code).
* Here the user enters a code that is assigned to every item. We have restricted ourselves to 10 items. In case the code is valid, the code will be sent to the database and the item name and price will be returned which will be displayed on GUI. All these processes happen in real time.
* [wx.grid.Grid](https://wxpython.org/Phoenix/docs/html/wx.grid.Grid.html#wx-grid-grid) and its related classes are used for displaying and editing tabular data.
* They provide a rich set of features for display, editing, and interacting with a variety of data sources. For simple applications, and to help you get started, [wx.grid.Grid](https://wxpython.org/Phoenix/docs/html/wx.grid.Grid.html" \l "wx-grid-grid) is the only class you need to refer to directly. It will set up default instances of the other classes and manage them for you. For more complex applications you can derive your own classes for custom grid views, grid data tables, cell editors and renderers.
* Further we are defining classes ‘MyGrid’ and ‘MyForm’ where we write other supporting functions for the working of GUI.