

**ITE1007- Object oriented analysis and design**

**Title: Near Field Communication**

Name: Sri Harshitha Palla

Reg. no: 17BIT0330

Contact no: 9949036116

Signature:

Name: Garlapati Saiteja

Reg. no: 17BIT0217

Contact no: 8985762222

Signature:

Name: Ranadeeshwar

Reg. no: 17BIT0133

Contact no: 9080183127

Signature:

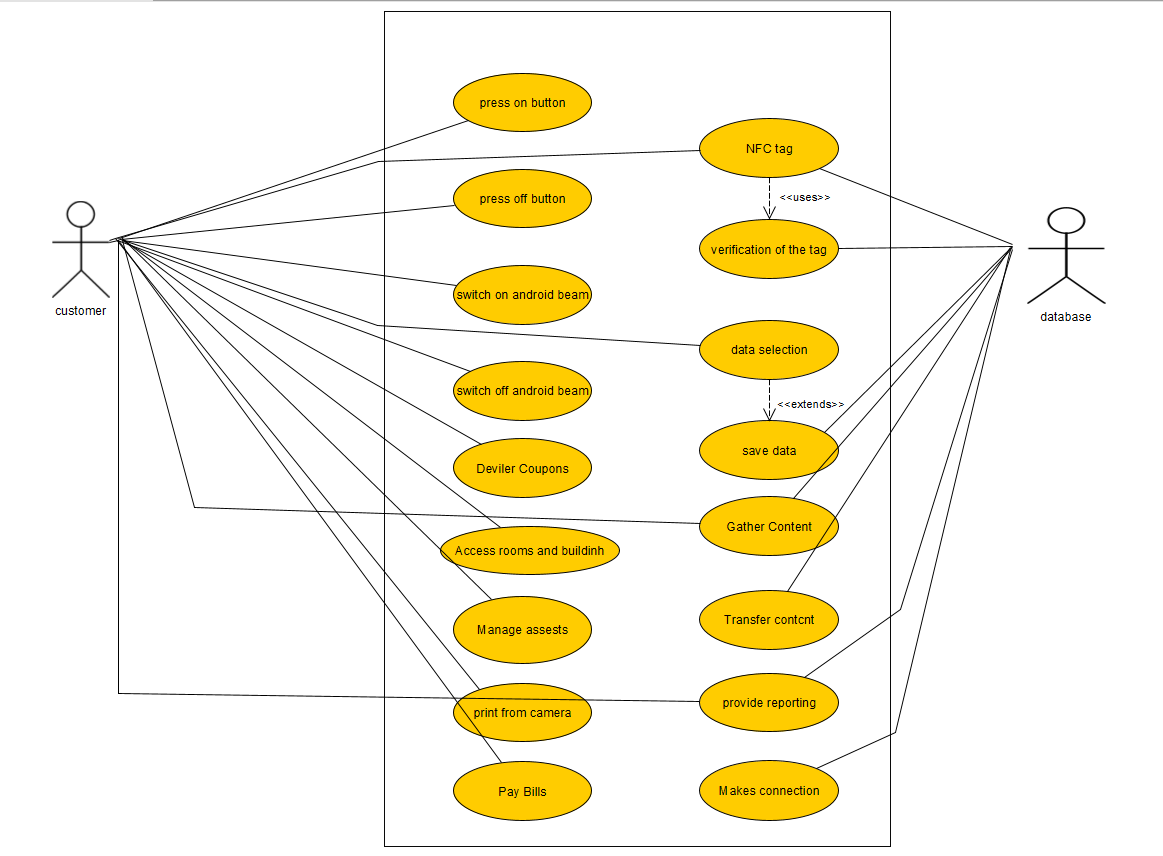
Problem Definition

Near Field Communication (NFC) is a special category or a case of Radio Frequency Identiﬁcation Technology. It is a wireless technology which provides communication between two mobile phones which contain a NFC tag, using short range radio waves. Like bluetooth it works just in short range and information exchange happens at low speed. The two gadgets can speak with one another utilizing NFC innovation when they contact one another or conveyed near one another. NFC also has an advantage over bluetooth that it provides bi-directional communication between devices, i.e., both devices can send and receive data simultaneously. Several security issues are as well attached with NFC which is a big concern. Security attacks like eavesdropping, data corruption and modification, interference attacks and theft, are the most dangerous for the customer who is using their smart phones for payment purpose.

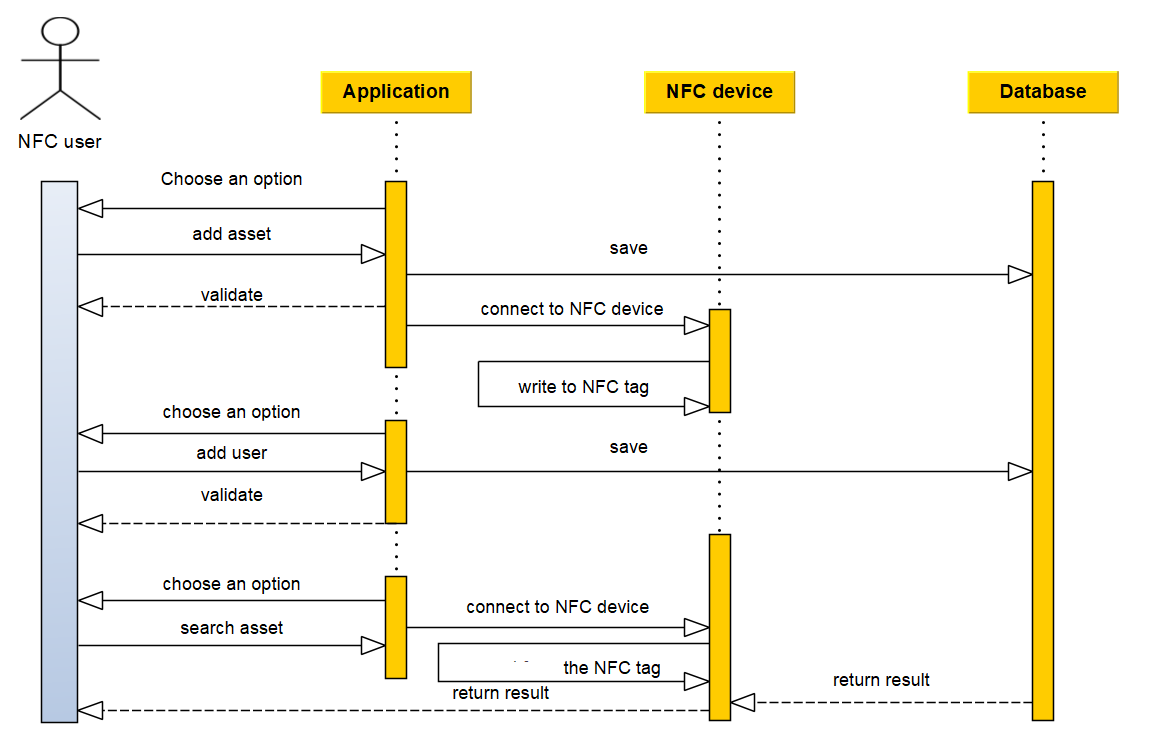
Software design tool used for the project

* yEd graph editor

USE CASE DIAGRAM

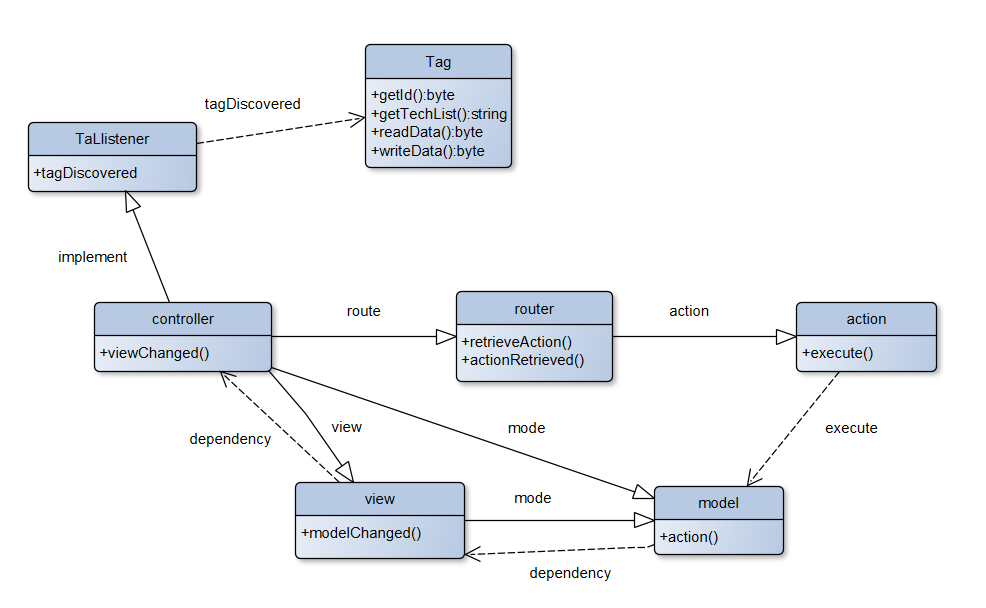


SEQUENCE DIAGRAM



CLASS DIAGRAM

This diagram shows the realization of the Model View Controller (MVC) software architecture pattern where the Controller realizes the TagListener interface that defines the tagDiscovered(tag)operation, which is executed when the mobile device NFC reader discovers an NFC tag. The Controllerdelegates to the Router the responsibility for finding the action (represented by an instance of the Actionclass) to be performed on the model. This action plays the role of Command in a realization of the Command design pattern where the receiver role is played by an instance of the Model class. As result of this execution, the model notifies its changes to its dependents by executing the modelChanged(aspect)operation. As View class instances depend on Model class instances, views receive notifications, which are forwarded to view dependents.



ACTIVITY DIAGRAM

