**WCOMSATS University Islamabad, Abbottabad Campus**

**Department of Computer Science**

**Project Proposal**

**Blood Donation Management System**

**CSC392 Object Oriented Software Engineering**

Submitted on: <Date&Time>

Group Members:

Waleed Khan (SP21-BSE-026)

…

Table of Contents

[CHAPTER 1 PROJECT PROPOSAL 3](#_Toc126512275)

[Introduction 3](#_Toc126512276)

[Vision and Business Case 3](#_Toc126512277)

[High level goals and concern 3](#_Toc126512278)

[Business case 3](#_Toc126512279)

[Executive Summary 3](#_Toc126512280)

[Use-Case Model 3](#_Toc126512281)

[Supplementary Specification 3](#_Toc126512282)

[Glossary 3](#_Toc126512283)

[Risk List & Risk Management Plan 3](#_Toc126512284)

[CHAPTER 2 USE CASES 4](#_Toc126512285)

[Use Case Diagram 4](#_Toc126512286)

[Use Cases Distribution 4](#_Toc126512287)

[Brief Level Use Cases 4](#_Toc126512288)

[Student Name 1 (Registration Number 1) 4](#_Toc126512289)

[Student Name 1 (Registration Number 1) 5](#_Toc126512290)

# CHAPTER 1 PROJECT PROPOSAL

## Introduction

BDMS (Blood donation management system) is a computerized application which is used to Manage the blood donation i.e.it handles blood donated to seeker, blood stored in blood bank and reward to the donor for donating the blood. It includes hardware components such as computer along with the software on which the system will run.

## Vision and Business Case

We envision a high quality fault-tolerant blood donation application, BDMS, with the flexibility to enable the donor to donate blood and the seeker to get blood, multiple terminal and user interface mechanisms, and integration with multiple third-party supporting system.

### the high-level goals and constraints

|  |  |  |  |
| --- | --- | --- | --- |
| High-level Goals | Priority | Problems and Concerns | Current Solutions |
| Fast, user friendly, blood management system | High | Website link down in case of high use.  Lack of up to date accurate information from database and other blood donation centers due to non synchronization |  |

### Business case

### Executive Summary

## Use-Case Model

Describes the functional requirements. During inception, the names of most use cases will be identified, and perhaps 10% of the use cases will be analyzed in detail.

## Supplementary Specification

***Describes other requirements, mostly non-functional. During inception, it is useful to have some idea of the key non-functional requirements that have will have a major impact on the architecture.***

[Text Book: 7.4. NextGen Example: (Partial) Supplementary Specification]

## Glossary

***Key domain terminology, and data dictionary.***

7.8. NextGen Example: A (Partial) Glossary

## Risk List & Risk Management Plan

***Describes the risks (business, technical, resource, schedule) and ideas for their mitigation or response.***

* + For example, leadership really wants a demo at the POSWorld trade show in Hamburg, in 18 months. But the effort for a demo cannot yet be even roughly estimated until deeper investigation.
  + For example, in the NextGen POS, for what remote services (e.g., tax calculator) will fail-over to local services be allowed? Why? Do they provide exactly the same services locally as remotely, or are there differences?

# CHAPTER 2 USE CASES

## Use Case Diagram

<Paste your semester project diagram imported from CASE Tool i.e. starUML here>

## Use Cases Distribution

|  |  |  |
| --- | --- | --- |
| S#. | Group Member | Assigned Use Cases |
| 1 | <Name>  <Reg no> | UC 1  UC 2 |
| … | … | … |

## Brief Level Use Cases

### Student Name 1 (Registration Number 1)

#### Use Case: Process Sale

A customer arrives at a checkout with items to purchase. The cashier uses the POS system to record each purchased item. The system presents a running total and line-item details. The customer enters payment information, which the system validates and records. The system updates inventory. The customer receives a receipt from the system and then leaves with the items.

Student Name 2 (Registration Number 2)

Student Name 3 (Registration Number 3)

Fully Dressed Use Cases

### Student Name 1 (Registration Number 1)

| Use Case UC1: Process Sale |
| --- |
| **Scope**: NextGen POS application  **Level**: user goal  **Primary** **Actor**: Cashier  **Stakeholders and Interests**:  - Cashier: Wants accurate, fast entry, and no payment errors, as cash drawer shortages are deducted from his/her salary.  - Salesperson: Wants sales commissions updated.  - Customer: Wants purchase and fast service with minimal effort. Wants easily visible display of entered items and prices. Wants proof of purchase to support returns.  - Company: Wants to accurately record transactions and satisfy customer interests. Wants to ensure that Payment Authorization Service payment receivables are recorded. Wants some fault tolerance to allow sales capture even if server components (e.g., remote credit validation) are unavailable. Wants automatic and fast update of accounting and inventory.  - Manager: Wants to be able to quickly perform override operations, and easily debug Cashier problems.  - Government Tax Agencies: Want to collect tax from every sale. May be multiple agencies, such as national, state, and county.  - Payment Authorization Service: Wants to receive digital authorization requests in the correct format and protocol. Wants to accurately account for their payables to the store.  **Preconditions**: Cashier is identified and authenticated. |

**Success Guarantee** (or Postconditions): Sale is saved. Tax is correctly calculated. Accounting and Inventory are updated. Commissions recorded. Receipt is generated. Payment authorization approvals are recorded.

**Main Success Scenario (or Basic Flow):**

1. Customer arrives at POS checkout with goods and/or services to purchase.
2. Cashier starts a new sale.
3. Cashier enters item identifier.
4. System records sale line item and presents item description, price, and running total. Price calculated from a set of price rules.

Cashier repeats steps 3-4 until indicates done.

1. System presents total with taxes calculated.
2. Cashier tells Customer the total, and asks for payment.
3. Customer pays and System handles payment.
4. System logs completed sale and sends sale and payment information to the external Accounting system (for accounting and commissions) and Inventory system (to update inventory).
5. System presents receipt.
6. Customer leaves with receipt and goods (if any).

**Extensions (or Alternative Flows):**

\*a. At any time, Manager requests an override operation:

1. System enters Manager-authorized mode.
2. Manager or Cashier performs one Manager-mode operation. e.g., cash balance change, resume a suspended sale on another register, void a sale, etc.
3. System reverts to Cashier-authorized mode.

\*b. At any time, System fails:

To support recovery and correct accounting, ensure all transaction sensitive state and events can be recovered from any step of the scenario.

1. Cashier restarts System, logs in, and requests recovery of prior state.
2. System reconstructs prior state.

2a. System detects anomalies preventing recovery:

* 1. System signals error to the Cashier, records the error, and enters a clean state.
  2. Cashier starts a new sale.

1a. Customer or Manager indicate to resume a suspended sale.

1. Cashier performs resume operation, and enters the ID to retrieve the sale.
2. System displays the state of the resumed sale, with subtotal.

2a. Sale not found.

* 1. System signals error to the Cashier.
  2. Cashier probably starts new sale and re-enters all items.

1. Cashier continues with sale (probably entering more items or handling payment).

**Special Requirements:**

- Touch screen UI on a large flat panel monitor. Text must be visible from 1 meter.

- Credit authorization response within 30 seconds 90% of the time.

- Somehow, we want robust recovery when access to remote services such the inventory system is failing.

- Language internationalization on the text displayed.

- Pluggable business rules to be insertable at steps 3 and 7.

- …

**Technology and Data Variations List**:

\*a. Manager override entered by swiping an override card through a card reader, or entering an authorization code via the keyboard.

3a. Item identifier entered by bar code laser scanner (if bar code is present) or keyboard.

3b. Item identifier may be any UPC, EAN, JAN, or SKU coding scheme.

7a. Credit account information entered by card reader or keyboard.

7b. Credit payment signature captured on paper receipt. But within two years, we predict many customers will want digital signature capture.

Frequency of Occurrence: Could be nearly continuous.

**Open Issues:**

- What are the tax law variations?

- Explore the remote service recovery issue.

- What customization is needed for different businesses?

- Must a cashier take their cash drawer when they log out?

- Can the customer directly use the card reader, or does the cashier have to do it?

**Screen Shots:**

<Paste the images from netbeans here for the relevant screen>