# User Story map

## Make a transaction

### Encounter the vending machine

Figure out what it does

### Decide what to pay

Count money – credit or cash

Look at product prices

### Purchase the product

Make a selection

Insert credit payment

Insert cash payment

Insert mobile payment

Receive change

### Receive the product

Watch the fun

Take the product

## Stock the machine

### Open machine

### Take stock actions

Insert stock

Set prices

## Maintain the machine

### Maintenance – Mechanical

Fix jams

Oil parts

### Maintenance – Cash

Take money out

Add change

### Get transaction logs

### Install machine

### Remove machine

# Use cases

## Actors

* Purchaser
* Maintenance

## Use cases

* Make a transaction
  + Make a transaction with credit
  + Make a transaction with cash
  + Make a transaction with mobile payment
* Stock machine
* Set product price
* Maintenance – Mechanical
  + Fix jams
  + Oil parts
* Maintenance – Cash
  + Take money out
  + Add change
  + Get transaction logs
* Install machine
* Remove machine

## Make a purchase with credit

#### Metadata

Information for project tracking

**Author** – AUCW class

**Date created** – 7/18/17

**Date revised** –

**Version** – 1.00

#### Use case data

Information about the use case

**Actors** – Purchaser, Maintenance

Does not include those participants involved but do not initiate/trigger this flow of events.

**Includes:**

**Extensions:**

**Related systems** – Credit card verification system

Sometimes called (supporting actors)

**System** – Vending machine system

Would not be used if this were a business use case

**Priority** –

Get this from the priority chart.

**Iteration** -

**Level** – Goal

Granularity options are: goal | partial goal | group of goals | group of partial goals

**Design constraints** –

Useful when you don’t use just business terms e.g. web site is required, SQL Server is required, location must be…

**Business goal** (value to sponsor) – increase revenue through larger market

#### Pre-conditions

Rules for beginning this use case: state of system prevents usage, must be testable

* Available communication to credit card processor
* at least one available product to sell
* jam mechanism detects no jams
* ~~Inside temperature is under 100 degrees F.~~ (no event in this course of events can trigger this, create separate system to push messages and see <<extends>>)

#### Course of Events

The sequence of tasks in conversation format between actor and system. Start each number with the system except the trigger and combine actor responses to system events.

1. The use case starts when the actor swipes credit card.
2. The system reads the card. The system validates the credit card.
   1. RULE – Brand of card – only MasterCard, VISA, and AmEx.
   2. RULE – Readable card – mag stripe reader is successful.
3. The system responds by prompting the actor to make a **product** (DD) selection ET#3. The actor makes a product selection (SD#1).
4. The system transmits a request to credit card processor to debit the credit card for the amount of the product price for credit. The credit card processor approves charge. The system logs the communication.
   1. RULE – Communication time-out – Maximum 10 seconds.
5. The system dispenses product. The system recognizes the product in the tray. The system sets the jam flag to true. The actor takes product.
6. The system checks for available products. The system sets the available products to vend flag to true.
7. The system recognizes that the actor took the product. The system sets the jam flag to false.
8. The system logs the transaction.
9. The system prompts with advertising welcome message (ET#4).

#### Extension points - optional

Extension points where use case can optionally go and then come back to the same point.

**Detect heat advisory** (#1-8) –The temperature system sent a message of high heat. Set the temperature flag to true.

**Detect no heat** **advisory** (#1-8) The temperature system sent a message of normal heat. Set the temperature flag to false.

#### Extension points – errors, exceptions

Errors occur at any point where there is a validation of a rule.

Errors occur at communication to other systems.

**Brand of card validation fails** (#3) – The system prompts the actor with a card failure message (ET#1)

**Readable card validation fails** (#3) – The system prompts the actor with a card failure message (ET#2)

**Jam occurs** (#5) – System did not recognize product in tray. The system prompts actor with … message. The system transmits a message to reverse the charge to the credit card processor. The system sets the jam flag to true.

**Product not taken** (5) -

**No available products** (6) - The system sets the available products to vend flag to false.

#### .Post-conditions

Is it really important to review the necessary outcomes of this use case? If so, summarize here. Also use for minimal conditions of satisfaction.

#### Notes/ Special Requirements

Any kind of quality, capacity, security, availability, disaster recovery information.

* **Security concern** – cards using chips may be more risky, check units and vendor recommendation.

### Data Dictionary

* **Product**
  + Price for cash
  + Price for credit
  + Name
* **Transaction**
  + **Product**
  + Quantity
  + Payment method
  + Jam flag status
* **Communication log**
  + Time/date
  + Product
  + Credit card without number
* **Credit card** 
  + Number
  + Exp date?
  + Name?

### Messages / External Text

ET#1 **Card failure** - MasterCard, VISA, or AmEx accepted.

ET#2 **Card failure** – Damaged card.

### Screen designs

SD#1 – Product selection panel

<screen shot>