It is a requirement to complete all three sections of Table below, and include the completed template as the first page of assignment that is submitted for marking.

Section I Reflecting on the feedback that I have received on previous assessments, the following issues/topics have been identified as areas for improvement: (add three bullet points). NB – for first-year students/PGTs in the first term, this refers to assessments in their previous institution. • It is recommended to add more practical examples to the lecture notes. • It is better to add some exercises to the handouts. • A concrete project can be taught in detailed.
Section II In this assignment, I have attempted to act on previous feedback in the following ways (3 bullet points) • Start looking up information and learning early. • Review the key aspects of the lecture notes again. • Make a clear format of the report.
Section III Feedback on the following aspects of this assignment (i.e. content/style/approach) would be particularly helpful to me: (3 bullet points) • The content of this assignment covers a wide range of UML modelling. • The style of this assignment is acceptable, and descriptions are in detailed. • The approach of this assignment is practical.

1. Introduction

1.1 Describe the EPS system requirements

The Electronic Prescription Service (EPS) system is designed to allow prescribers to send prescriptions electronically to a dispenser of the patient's choice.

The EPS system shall provide prescribers with the operation of logging onto the clinical system and sending electronic prescriptions.

The EPS system shall enable patients to log on by using the NHS Smartcard and provide patients with the operation of obtaining prescriptions.

The EPS system shall allow the dispensers to complete the dispensing process.

The EPS system shall have a means of supporting the reimbursement claim process.

1.2 State the user requirements

Users are defined as patients, dispensers, prescribers and the reimbursement agency in this use-case modelling. Their requirements are verified and validated as follows.

The patients may access to the EPS, nominate dispensing contractor and receive prescriptions and messages from EPS.

The prescribers may log onto the clinical system, choose a medical appliance for the patients, add prescribing endorsements, apply electronic

signature and transmit the electronic prescriptions to the EPS.

The dispensers may retrieve electronic prescriptions from the EPS, dispense them to patients, record the status of each of the prescription items, send dispense notification to the EPS and submit reimbursement endorsement messages to the reimbursement agency.

The reimbursement agency can make a payment.

2. Use-Case Diagram, Survey description and CRC cards

2.1 Create the Use-Case diagram

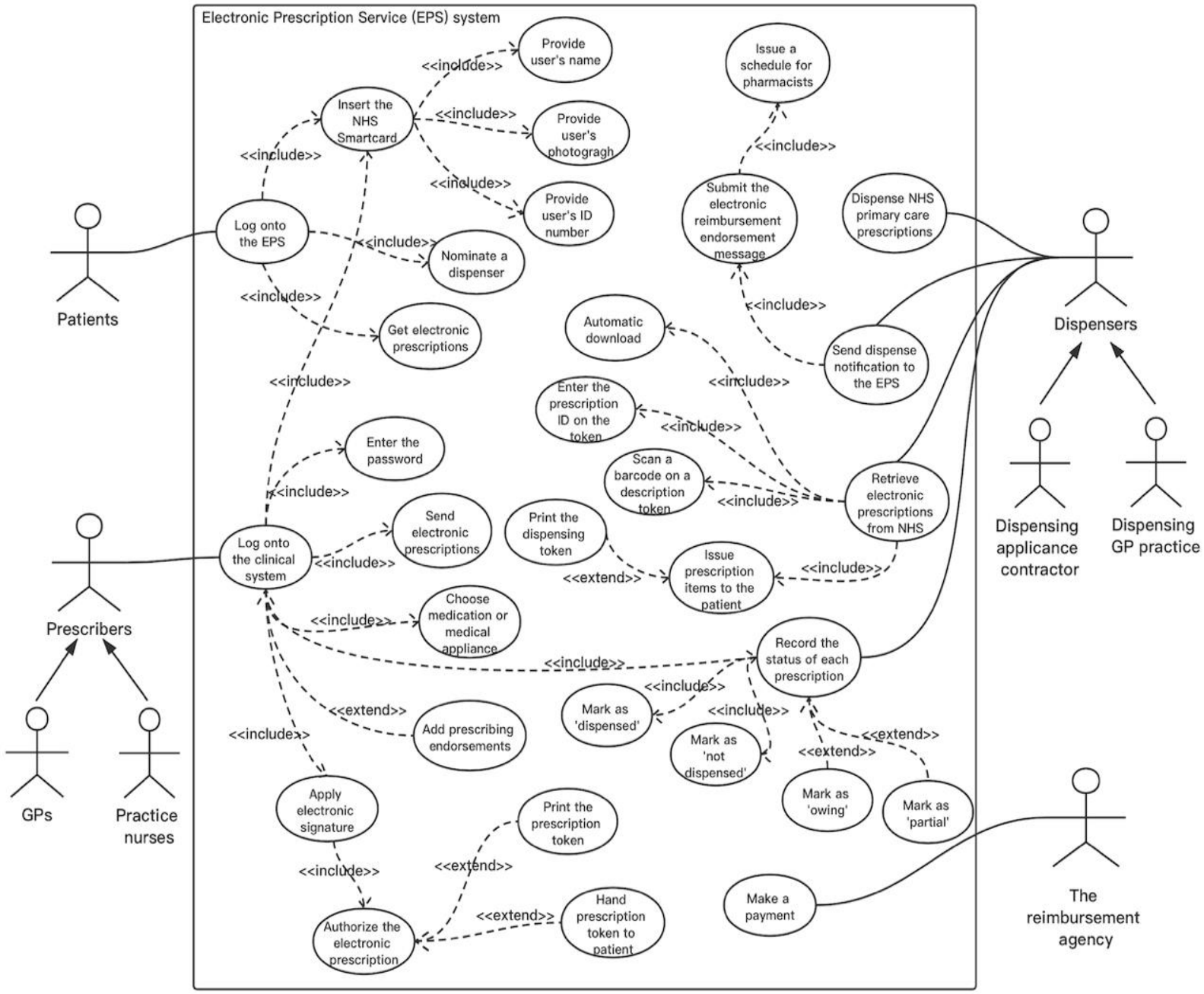

Figure 1. The Use-Case Diagram

2.2 Describe the Use-Case scenario (Use-Case Survey Description)

The patients may log onto the EPS by inserting or clustering the NHS Smartcard having their name, photograph, ID number. Then, they can nominate a dispensing contractor and receive electronic prescriptions

included NHS primary care prescriptions and messages from EPS.

The prescribers may log onto the clinical system by using their NHS Smartcard and entering the password. Then, they can choose a medical appliance for the patients, add prescribing endorsements where required, apply an electronic signature to authorize the electronic prescription and transmit the electronic prescriptions to the EPS.

The dispensers may dispense NHS primary care prescriptions to patients, send dispense notification to the EPS included submitting reimbursement endorsement messages to the reimbursement agency and issuing a schedule for pharmacists. They can also retrieve electronic prescriptions from the EPS and record the status of each of the prescription items.

The reimbursement agency can make a payment.

2.3 Identify classes

2.3.1 Class identification

a) Patient scenario: logon, smartcard, prescription.

Logon is a vital class identifier. Smartcard is a class identifier that can contain variables such as name, ID, photograph.

b) Prescriber scenario: logon, prescription, prescription token.

c) Dispenser scenario: dispensing token, message, record, schedule.

‘Prescription or dispensing token’ is likely generalized to one class called ‘token’.

d) Reimbursement agency scenario: payment.

e) An initial class list: logon, smartcard, prescription, token, payment, message, record, schedule.

2.3.2 Stereotypical classes

a) Entity: patient, prescriber, dispenser, reimbursement agency, token, prescription, NHS Smartcard.

b) Control: send information, record status, access system, make payment.

c) Boundary: menu, window, dialogBox.

d)The updated initial class list: patient, prescriber, dispenser, reimbursement agency, prescription.

2.4 Write Class-Responsibilities-Collaboration cards for each class

Table 1. CRC cards

Patient	
Responsibilities	**Collaborators**
Patients nominate a dispenser and get electronic prescriptions sent by a prescriber. During this process, the patient logs onto the EPS system and get electronic prescriptions.	Prescriber Dispenser Prescription

Prescriber	
Responsibilities	**Collaborators**
Prescribers log onto the clinical system, apply an electronic signature to authorize the electronic prescription, add prescribing endorsement.	Prescription Dispenser

å

Dispenser	
Responsibilities	**Collaborators**
Dispensers dispense NHS prescriptions to patients, send dispense notification to the EPS, retrieve	Prescriber Patient Reimbursement agency

electronic prescriptions from NHS, record the status of each prescription.	Prescription

Reimbursement agency	
Responsibilities	**Collaborators**
Reimbursement agency class makes a payment if it receives the message sent by the dispenser.	Dispenser

Prescription	
Responsibilities	**Collaborators**
Prescribers send electronic prescriptions, and dispensers dispense them to patients.	Prescriber Dispenser Patients

3. Class diagram

3.1 Attribute identification

a) Patient: namePatient, photoPatient, IdPatient.

b) Prescriber: namePrescriber, photoPrescriber, IdPrescriber.

c) Dispenser: nameDispenser, itemMessage.

d) Reimbursement agency: amoutPayment, productName.

e) Prescription: nameItem, schedulDate.

3.2 Method identification:

a) Patient: insertSmartcard(); getPrescriptions().

b) Prescriber: insertSmartcard(); enterPassword(); applySignature(); ChooseMedication().

c) Dispenser: dispensePrescriptions(); retrievePrescriptions(); submitMessage(); recordStatus(); sendPrescriptions(); sendNotification(); issueSchedule(); issueItems();

d) Reimbursement agency: makePayment();

e) Prescription: prescriptionToken(); schedulEndorsement().

3.3 Detailed the Class Diagram

In Figure 2, it shows that attributes, methods, their visibilities, arguments, return types and relations to define architecture.

The visibility of attributes in class 'Prescriber' and 'Patient' should be protected. The visibilities of other attributes can be public.

'Dispenser' class has an association relationship with 'Prescriber' class,

'Patient' class and 'Prescription' class. It can define connections between multiple items.

'Prescription' class has a dependency relationship with 'Prescriber' class. It won't be very sensible if prescribers don't create a prescription. The methods in the 'Prescription' class are valid based on a dependency relationship. 'Reimbursement Agency' class has a dependency relationship with 'Dispenser' class because only if the reimbursement agency receives the message of a dispenser, it can make a payment.

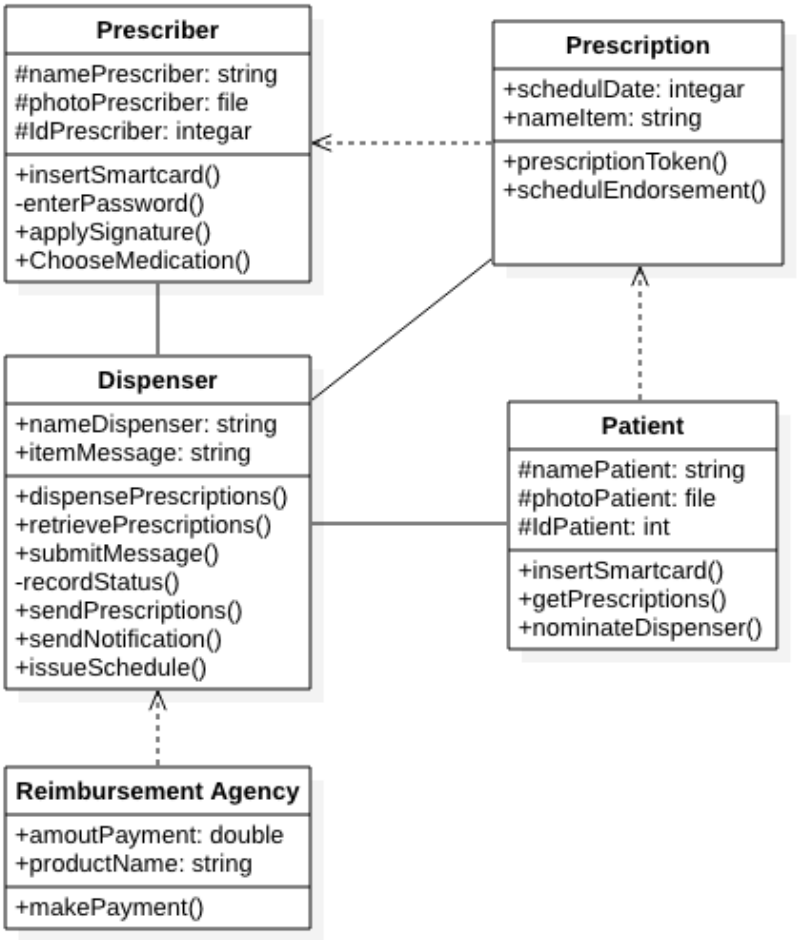

Figure 2. The Class Diagram

4. Interaction diagram

4.1 Collaboration diagram

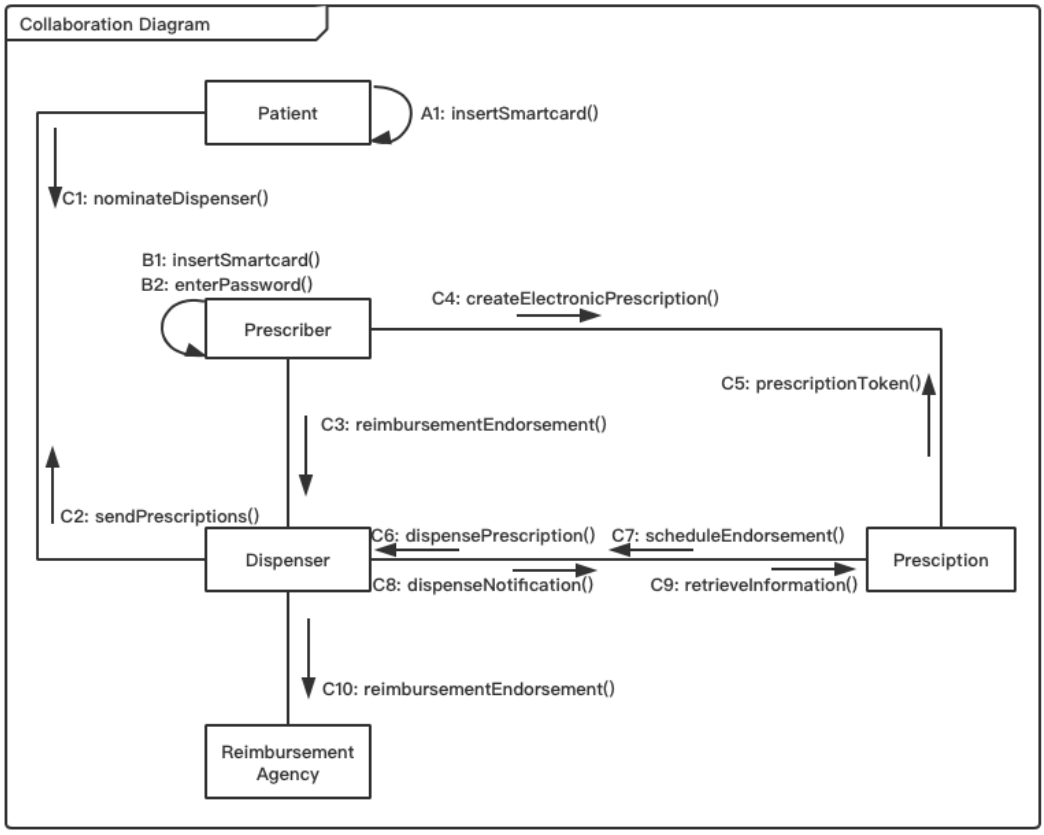

Figure 3. Collaboration diagram

The EPS system above is realized by five classes which work together as described by collaborations to achieve these as follows.

After patients and prescribers log onto the system, patients can nominate a dispenser contractor. Prescribers create electronic prescriptions. Dispensers retrieve and dispense them to patients. Then, the reimbursement agency receives the message to make a payment. Besides, dispensers schedule endorsement. The prescription and dispensing token can be printed if required. Prescription items are issued to the patient or patient's representative.

4.2 Sequence diagram

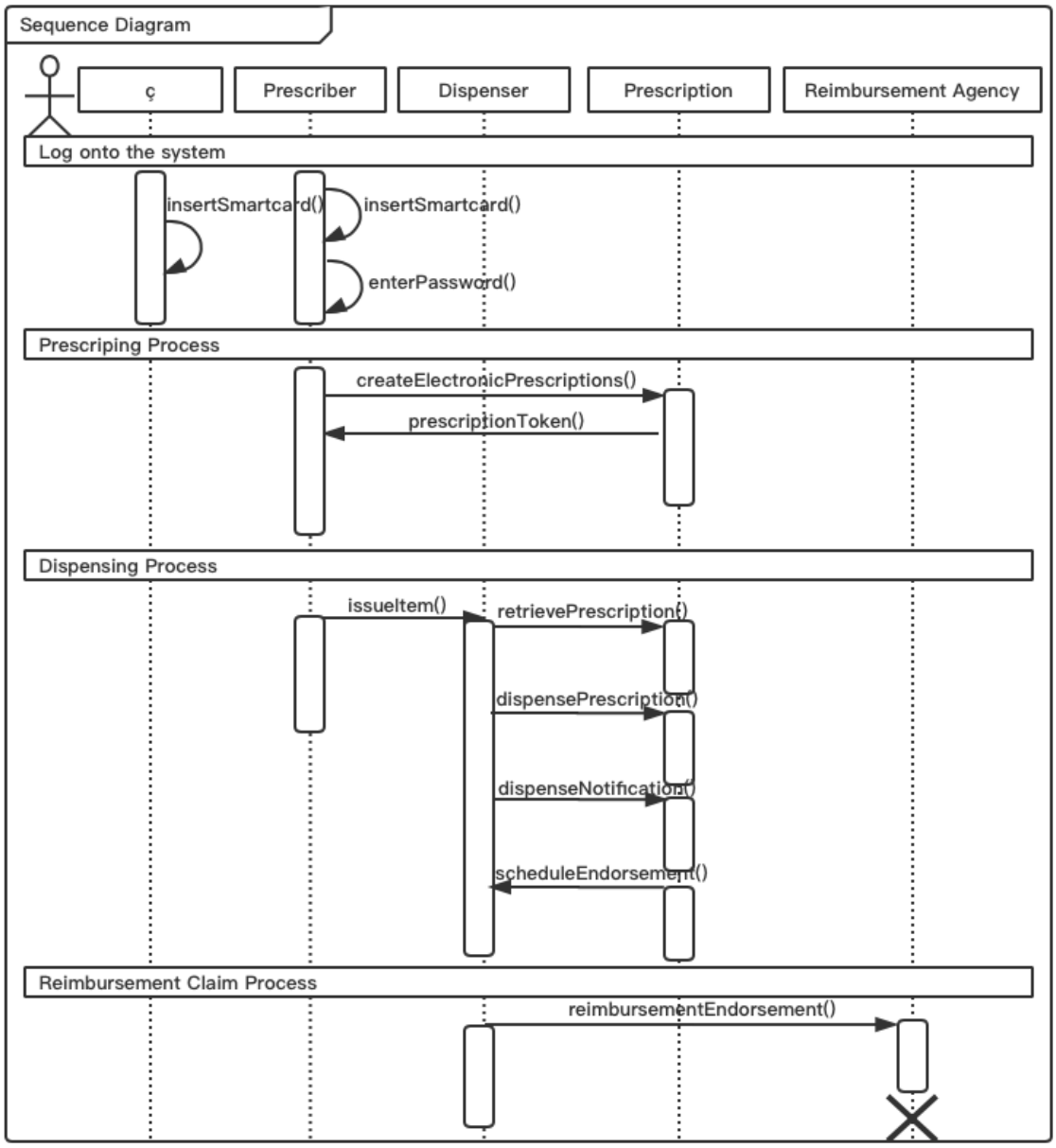

Figure 4. The Sequence Diagram

We have to consider message transmissions between classes in the sequence diagram shown above and minimize the number of relationships.

5. State-Chart diagram

First, patients insert their NHS Smartcard to log onto the system. Prescribers access to the clinical system by using their NHS Smartcard and entering the password.

Second, prescribers choose medical appliance, apply electronic signature, add prescribing endorsement to complete sending electronic prescriptions to patients.

Third, dispensers retrieve prescriptions and record the status of them in order to execute dispensing.

Then, after submitting reimbursement endorsement message and sending dispense notifications, the reimbursement agency makes a payment.

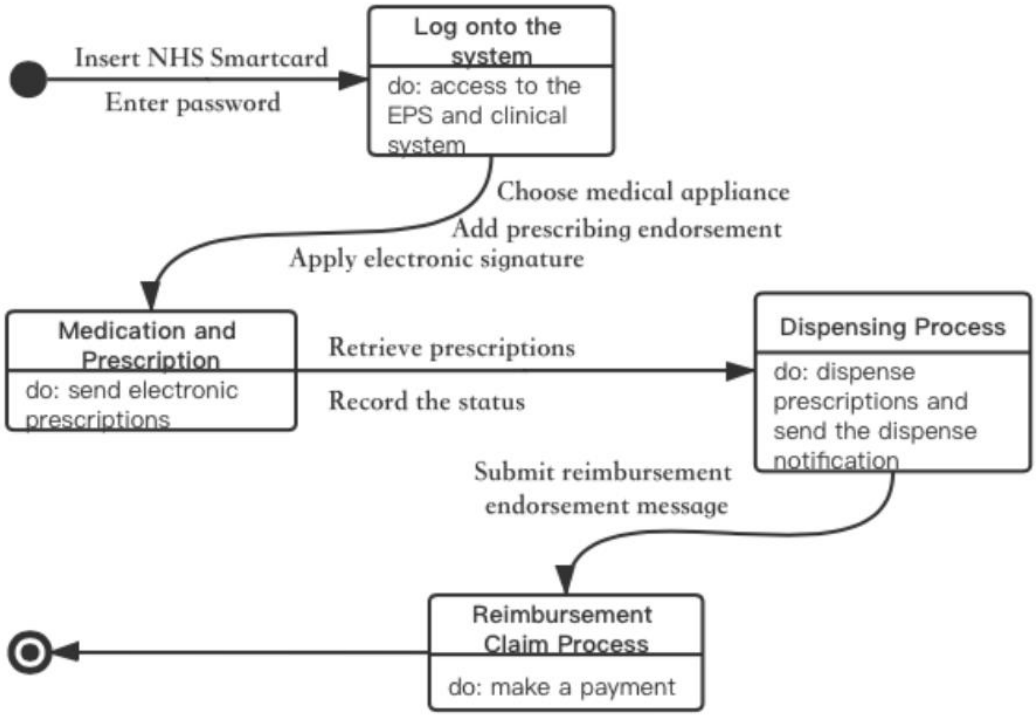

Figure 5. The State Chart Diagram

6. Discussion of non-functional requirements

It is of paramount importance to consider environmental and implementation constraints associated with each use-case.

First, the speed of response should be fast while the EPS system is running. Otherwise, prescribers cannot choose medication to patients on time, patients cannot be treated on time, and medical expenses are not reimbursed as soon as possible.

Second, the EPS system should effectively support concurrent programming to implement platform independence.

Third, it has to increase the maintainability and extensibility of the EPS system to consider future user growth and feature additions.

Last but not least, the EPS system must have perfect reliability; otherwise, the patient's medical privacy information will be leaked.

7. Produce pseudocode for major functions

```
class Patient:
    def __init__(self):
        self.namePatient = None
        self.photoPatient = None
        self.IdPatient = None

    def insertSmartcard(self, ):
        pass

    def getPrescriptions(self, ):
        pass

    def nominateDispenser(self, ):
        pass
class Prescriber:
    def __init__(self):
        self.namePrescriber = None
        self.photoPrescriber = None
        self.IdPrescriber = None

    def insertSmartcard(self, ):
        pass

    def enterPassword(self, ):
        pass

    def applySignature(self, ):
        pass

    def ChooseMedication(self, ):
        pass
class Dispenser:
    def __init__(self):
        self.nameDispenser = None
        self.itemMessage = None

    def dispensePrescriptions(self, ):
        pass

    def retrievePrescriptions(self, ):
        pass
```

```
    def submitMessage(self, ):
        pass

    def recordStatus(self, ):
        pass

    def sendPrescriptions(self, ):
        pass

    def sendNotification(self, ):
        pass

    def issueSchedule(self, ):
        pass
class Reimbursement Agency:
    def __init__(self):
        self.amoutPayment = None
        self.productName = None

    def makePayment(self, ):
        pass
class Prescription:
    def __init__(self):
        self.schedulDate = None
        self.nameItem = None

    def prescriptionToken(self, ):
        pass

    def schedulEndorsement(self, ):
        pass
```