```
──────────────────── MODULE juniorv1 ────────────────────
Redeclaration of specdatamodels variables

EXTENDS Sequences, Naturals, FiniteSets

CONSTANT USERS

CONSTANTS
    SubscriptionFee,
    CancellationFee,
    FailedPaymentFee

VARIABLES
    month,
    database,
    events

vars ≜ ⟨events, month, database⟩

INSTANCE specdatamodels

Now ≜ Len(events)

Months ≜ 0 .. 10

Strong Typing

Month ≜ Nat

Database Rows

UserRow ≜ [
    subscribed : BOOLEAN ,
     Forget canceled
    inTrial : BOOLEAN ,
    trialStartTime  : Nat,
    billedForMonth : Nat
]

BillQueueItem ≜ [
    user : USERS,
    fee : Fees
]

TypeOk ≜
    ∧  EventsOk
    ∧  month ∈ Month
    ∧  database.users ∈ [USERS → UserRow]
    $\land \quad database.billQueue \in Seq(BillQueueItem)$
```

API endpoints

```
$StartSubscription(u) \triangleq$
    $\land\ database.users[u].subscribed = \text{FALSE}$
    $\land\ database' = [database\ \text{EXCEPT}$
                    $![\text{“users”}][u].subscribed = \text{TRUE}$
                $]$
     Observability required by stub
    $\land\ events' = Append(events, [type \mapsto \text{“startsubscription”}, user \mapsto u])$
    $\land\ \text{UNCHANGED}\ month$

$CancelSubscription(u) \triangleq$
    $\land\ database.users[u].subscribed = \text{TRUE}$
    $\land\ database' =$
        $[database\ \text{EXCEPT}$
            $![\text{“users”}][u].subscribed = \text{FALSE},$
             Charge cancellation fee
            $![\text{“billQueue”}] =$
                $Append(database.billQueue,$
                    $[user \mapsto u, fee \mapsto CancellationFee])$
        $]$

     Observability required by stub
    $\land\ events' = Append(events, [type \mapsto \text{“cancelsubscription”}, user \mapsto u])$
    $\land\ \text{UNCHANGED}\ \langle month \rangle$

$StartTrial(u) \triangleq$
    $\land\ database.users[u].inTrial = \text{FALSE}$
    $\land\ database.users[u].subscribed = \text{FALSE}$
    $\land\ database' = [database\ \text{EXCEPT}$
                    $![\text{“users”}][u].inTrial = \text{TRUE}]$

     Observability required by stub
    $\land\ events' = Append(events, [type \mapsto \text{“starttrial”}, user \mapsto u])$
    $\land\ \text{UNCHANGED}\ \langle month \rangle$

$CancelTrial(u) \triangleq$
    $\land\ database.users[u].inTrial = \text{TRUE}$
    $\land\ database' = [database\ \text{EXCEPT}$
                    $![\text{“users”}][u].inTrial = \text{FALSE}$
                $]$
    Observability required by stub
    $\land\ events' = Append(events,\ [type \mapsto \text{"canceltrial"},\ user \mapsto u])$
    $\land$ UNCHANGED $\langle month \rangle$

$WatchVideo(u) \triangleq$
    $\land\ \lor\ database.users[u].subscribed = \text{TRUE}$
       $\lor\ database.users[u].inTrial = \text{TRUE}$

    Observability required by stub
    $\land\ events' = Append(events,\ [type \mapsto \text{"watchvideo"},\ user \mapsto u])$
    $\land$ UNCHANGED $\langle month,\ database \rangle$

 Stub method, do not change
$Bill(u,\ fee) \triangleq$
    $\land\ events' = Append(events,\ [type \mapsto \text{"bill"},$
                                       $user \mapsto u,$
                                       $fee \mapsto fee])$

$PaymentFailed(u,\ fee) \triangleq$
    $\land\ database' = [database$ EXCEPT
                         $![\text{"users"}][u].subscribed = \text{FALSE}$
                     $]$

    Observability required by stub
    $\land\ events' = Append(events,\ [type \mapsto \text{"paymentfailed"},$
                                       $user \mapsto u,$
                                       $fee \mapsto fee])$
    $\land$ UNCHANGED $\langle month \rangle$

Recurring Operations

 This the the state that calls the Payment Failed API
$ExistingBillFailed \triangleq$
    $\lor\ \exists\, i \in 1 \,..\, Len(events):$
          Only a past bill can fail
        $\land\ events[i] \in BillEvent$
        $\land\ PaymentFailed(events[i].user,\ events[i].fee)$

$BillSubscribedUsers \triangleq$
    $\exists\, u \in USERS:$
        That is subscribed
        $\land\ \lor\ database.users[u].subscribed = \text{TRUE}$
            Subscribed from a trial so bill
           $\lor\ \land\ database.users[u].inTrial = \text{TRUE}$
              $\land\ database.users[u].trialStartTime < month$
        Ensure users are not double billed
```

```
          ∧ database.users[u].billedForMonth < month
          ∧ database′ =
             [database EXCEPT
                  Add subscription fee
                 ![“billQueue”] =
                     Append(database.billQueue,
                                [user ↦ u, fee ↦ SubscriptionFee]),
                 ![“users”][u].billedForMonth = month
             ]

ProcessBills ≜
      ∧ Len(database.billQueue) > 0
      ∧ LET bill ≜ Head(database.billQueue) IN
           Bills user
           ∧ Bill(bill.user, bill.fee)
           ∧ database′ =
              [database EXCEPT
                   Removes head of queue
                  ![“billQueue”] =
                      SubSeq(database.billQueue,
                      2, Len(database.billQueue))
              ]

Stub method that prevents the month from passing until all operations are complete. Represent
worker methods, etc.
HandledMonth ≜
      ∧ ¬ENABLED BillSubscribedUsers
      ∧ ¬ENABLED ProcessBills

 DO NOT MODIFY
MonthPasses ≜
      ∧ HandledMonth
      ∧ month′ = month + 1
      ∧ events′ = Append(events, [type ↦ “monthpass”])
      ∧ UNCHANGED ⟨database⟩

Specification
Init ≜
      ∧ events = ⟨⟩  Events must be initialized empty, per stub
      ∧ month = 0
      ∧ database = [
                Users start with everything unset
             users ↦
                 [u ∈ USERS ↦
                    [
                     subscribed ↦ FALSE,
                  inTrial ↦ FALSE,
                  trialStartTime  ↦ 0,
                  billedForMonth ↦ 0
                 ]
               ],
              Bill queue starts empty
             billQueue ↦ ⟨⟩
         ]

Next ≜
   \* Required by stub
   ∨ MonthPasses
   \* State modified below
   ∨ ∃ u ∈ USERS:
        ∨ StartSubscription(u)
        ∨ CancelSubscription(u)
        ∨ StartTrial(u)
        ∨ CancelTrial(u)
        ∨ WatchVideo(u)
        \* Add more user based states
   \* Payment failing behavior is part of spec not implementation
   ∨ ExistingBillFailed
   ∨ BillSubscribedUsers

 \* Modification History
 \* Last modified Sun Jun 19 17:39:21 MST 2022 by elliotswart
 \* Created Fri Jun 17 00:43:20 MST 2022 by elliotswart
```