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

EXTENDS Sequences, Naturals, FiniteSets

 Represents every potential user in the system
CONSTANT USERS

 Constants that should be set to single model values, to allow comparisons
 Only equality comparisons will be made.
CONSTANTS
    SubscriptionFee,
    CancellationFee,
    FailedPaymentFee

VARIABLES
     Represents the current month
    month,
     Represents the status of the database. Design requirements require
     that all persistant application state be stored here
    database,
     Required by spec
    events

vars ≜ ⟨events, month, database⟩

 Provides all the data models required by the spec
INSTANCE specdatamodels

Now ≜ Len(events)

Months ≜ 0 . . 10

Strong Typing

Month ≜ Nat

Database Rows

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

BillQueueItem ≜ [
    $user : USERS,$
    $fee : Fees$
$]$

$TypeOk \triangleq$
    $\land\ EventsOk$
    $\land\ month \in Month$
    $\land\ database.users \in [USERS \rightarrow UserRow]$
    $\land\ database.billQueue \in Seq(BillQueueItem)$
```

API endpoints

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

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

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

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

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

$CancelTrial(u) \triangleq$
    $\land$ $database.users[u].inTrial = $ TRUE
    $\land$ $database' = [database$ EXCEPT
                    $![\text{"users"}][u].inTrial = $ 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 = $ TRUE
      $\lor$ $database.users[u].inTrial = $ 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 changed
$Bill(u, fee) \triangleq$
    $\land$ $events' = Append(events, [type \mapsto \text{"bill"},$
                                   $user \mapsto u,$
                                   $fee \mapsto fee])$

$PaymentFailed(u, fee) \triangleq$
    $\land$ $database' = [database$ EXCEPT
                    $![\text{"users"}][u].subscribed = $ 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
        $\land\ database.users[u].billedForMonth < month$
        $\land\ database' =$
            $[database\ \text{EXCEPT}$
                Add subscription fee
                $![\text{"billQueue"}] =$
                    $Append(database.billQueue,$
                        $[user \mapsto u,\ fee \mapsto SubscriptionFee]),$
                $![\text{"users"}][u].billedForMonth = month$
            $]$

$ProcessBills \triangleq$
    $\land\ Len(database.billQueue) > 0$
    $\land\ \text{LET}\ bill \triangleq Head(database.billQueue)\text{IN}$
        Bills user
        $\land\ Bill(bill.user,\ bill.fee)$
        $\land\ database' =$
            $[database\ \text{EXCEPT}$
                Removes head of queue
                $![\text{"billQueue"}] =$
                    $SubSeq(database.billQueue,$
                    $2,\ Len(database.billQueue))$
            $]$
```

Stub method that prevents the month from passing until all operations are complete. Represent worker methods, etc

```
$HandledMonth \triangleq$
    $\land\ \neg\text{ENABLED}\ BillSubscribedUsers$
    $\land\ \neg\text{ENABLED}\ ProcessBills$
```

DO NOT MODIFY

```
$MonthPasses \triangleq$
    $\land\ HandledMonth$
    $\land\ month' = month + 1$
    $\land\ events' = Append(events,\ [type \mapsto \text{"monthpass"}])$
    $\land\ \text{UNCHANGED}\ \langle database \rangle$
```

Specification

```
Init ≜
    ∧ events = ⟨⟩  Events must be intialized 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
```