MODULE *stubs*

Redeclaration of *specdatamodels* variables

EXTENDS *Sequences*, *Naturals*, *FiniteSets*

CONSTANT *USERS*

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

$vars \triangleq \langle events,\ month,\ database \rangle$

Provides all the data models required by the spec
INSTANCE *specdatamodels*

$Now \triangleq Len(events)$

$Months \triangleq 0\,..\,10$

Strong Typing

$Month \triangleq Nat$

Database Rows

$TypeOk \triangleq$
$\quad \land\ EventsOk$
$\quad \land\ month \in Month$
Additional type definitions

API endpoints

$StartSubscription(u) \triangleq$
Add logic here

Observability required by stub

```
    $\land\ events' = Append(events,\ [type \mapsto \text{“startsubscription”},\ user \mapsto u])$
    $\land$ UNCHANGED $month$

$CancelSubscription(u)\ \triangleq$
    Add logic here

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

$StartTrial(u)\ \triangleq$
    Add logic here

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

$CancelTrial(u)\ \triangleq$
    Add logic here

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

$WatchVideo(u)\ \triangleq$
    Add logic here

    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
$Bill(u,\ fee)\ \triangleq$
    $\land\ events' = Append(events,\ [type \mapsto \text{“bill”},$
                                     $user \mapsto u,$
                                     $fee \mapsto fee])$

$PaymentFailed(u,\ fee)\ \triangleq$
    Add logic here

    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
        $\land events[i] \in BillEvent$
        $\land PaymentFailed(events[i].user, \; events[i].fee)$
```

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

```
$HandledMonth \triangleq$
    Replace logic here
    $\land True$
```

```
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

```
$Init \triangleq$
    $\land events = \langle \rangle$    Events must be initialized empty, per stub
    $\land month = 0$
    $\land database = \square$   Add record here

$Next \triangleq$
    Required by stub
    $\lor MonthPasses$
    State modified below
    $\lor \exists u \in USERS :$
        $\lor StartSubscription(u)$
        $\lor CancelSubscription(u)$
        $\lor StartTrial(u)$
        $\lor CancelTrial(u)$
        $\lor WatchVideo(u)$
         Add more user based states
    Payment failing behavior is part of spec not implementation
    $\lor ExistingBillFailed$
    Add more global states
```

* Modification History

\ * Last modified Sun *Jun* 19 15:35:00 *MST* 2022 by *elliotswart*
\ * Created *Thu* *Jun* 16 19:34:32 *MST* 2022 by *elliotswart*