section *homestay* **parents** *standard_toolkit*

Contents

1 General

section *general* **parents** *homestay*

We want to have a few basic types here. We need to know general information about all of the users in the system:

- Email address, this is the unique identifier in the system

- Password, this has restrictions:

    - At least 8 characters in length
    - 1 digit
    - 1 uppercase character
    - 1 lowercase character

- First name, should be non-empty

- Last name, should be non-empty

$$[Email, Password, FirstName, LastName]$$

$$
\begin{aligned}
Month ::=\ & January \\
& |\ February \\
& |\ March \\
& |\ April \\
& |\ May \\
& |\ June \\
& |\ July \\
& |\ August \\
& |\ September \\
& |\ October \\
& |\ November \\
& |\ December \\
Day ==\ & \mathbb{N} \\
Year ==\ & \mathbb{N}
\end{aligned}
$$

Using these basic types we construct more complex types in the system. The applicant is the user of the system. Password tokens are used to reset the password.

$$
\begin{aligned}
Applicant &== Email \times Password \times FirstName \times LastName \\
PToken &== \mathbb{N} \times Email \\
Date &== Month \times Day \times Year
\end{aligned}
$$

We need to be able to notify the user of what is happening in the application. So, we enumerate the possible responses from actions taken.

$Response$ ::=
    $InvalidToken$
    $\mid PasswordResetSuccessful$
    $\mid LoginSuccessful$
    $\mid InvalidAvailability$
    $\mid ValidAvailability$

Our initial state is just a bunch of empty sets.

```
┌─ HomestayInitial ──────────────────────────
│  Applicants : ℙ Applicant
│  Emails : ℙ Email
│  Passwords : ℙ Password
│  Valid, Invalid : ℙ PToken
├────────────────────────────────────────────
│  Applicants = ∅
│  Emails = ∅
│  Passwords = ∅
│  Valid = Invalid = ∅
└────────────────────────────────────────────
```

In the database we need to ensure that there is each email address is unique, and that the password tokens are either valid or invalid.

```
┌─ HomestayDatabase ─────────────────────────
│  HomestayInitial
├────────────────────────────────────────────
│  ∀ e₀, e₁ : Email; p₀, p₁ : Password; fn₀, fn₁ : FirstName; ln₀, ln₁ : LastName •
│      e₀ = e₁ ⇒ (p₀ = p₁ ∧ fn₀ = fn₁ ∧ ln₀ = ln₁)
│  Valid ∩ Invalid = ∅
└────────────────────────────────────────────
```

1.1 General Operators

It gets kind of silly to have to rewrite these operators and functions each time, so we have some helpers here.

We want an easy way to update an applicant. This helper function updates the applicant in the set of all applicants, identified by the email address.

function 42 **leftassoc** $(\_ \otimes \_)$

$$\_ \otimes \_ : \mathbb{P}\, Applicant \times Applicant \to \mathbb{P}\, Applicant$$

$$
\begin{array}{l}
\forall\, a_1 : Applicant;\ as : \mathbb{P}\, Applicant \bullet \\
\quad \exists\, p_0, p_1 : Password; \\
\quad fn_0, fn_1 : FirstName; \\
\quad ln_0, ln_1 : LastName; \\
\quad e : Email; \\
\quad a_0 : Applicant \mid \\
\qquad a_1 = (e, p_1, fn_1, ln_1) \wedge (a_0 = (e, p_0, fn_0, ln_0) \in as) \bullet \\
\qquad\quad as \otimes a_1 = (as \setminus \{a_0\}) \cup \{a_1\}
\end{array}
$$

[X, Y, Z]

$$
\begin{array}{l}
firstOf3 == \lambda\, x : X;\ y : Y;\ z : Z \bullet x \\
secondOf3 == \lambda\, x : X;\ y : Y;\ z : Z \bullet y \\
thirdOf3 == \lambda\, x : X;\ y : Y;\ z : Z \bullet z
\end{array}
$$

[X, Y, Z]

$$updateFirstOf3 : (X \times Y \times Z) \times X \to X \times Y \times Z$$

$$
\begin{array}{l}
\forall\, xyz : X \times Y \times Z;\ x : X \bullet \\
\quad \exists\, x_0 : X;\ y : Y;\ z : Z \mid xyz = (x_0, y, z) \bullet \\
\qquad updateFirstOf3(xyz, x) = (x, y, z)
\end{array}
$$

[X, Y, Z]

$$updateSecondOf3 : (X \times Y \times Z) \times Y \to X \times Y \times Z$$

$$
\begin{array}{l}
\forall\, xyz : X \times Y \times Z;\ y : Y \bullet \\
\quad \exists\, x : X;\ y_0 : Y;\ z : Z \mid xyz = (x, y_0, z) \bullet \\
\qquad updateSecondOf3(xyz, y) = (x, y, z)
\end{array}
$$

[X, Y, Z]

$$updateThirdOf3 : (X \times Y \times Z) \times Z \to X \times Y \times Z$$

$$
\begin{array}{l}
\forall\, xyz : X \times Y \times Z;\ z : Z \bullet \\
\quad \exists\, x : X;\ y : Y;\ z_0 : Z \mid xyz = (x, y, z_0) \bullet \\
\qquad updateThirdOf3(xyz, z) = (x, y, z)
\end{array}
$$

We're going to Greenspun up some stuff.

$$Ord ::= LT \mid EQ \mid GT$$

$month2Nat : Month \rightarrow \mathbb{N}$

$month2Nat\ January = 1\ \wedge$
$month2Nat\ February = 2\ \wedge$
$month2Nat\ March = 3\ \wedge$
$month2Nat\ April = 4\ \wedge$
$month2Nat\ May = 5\ \wedge$
$month2Nat\ June = 6\ \wedge$
$month2Nat\ July = 7\ \wedge$
$month2Nat\ August = 8\ \wedge$
$month2Nat\ September = 9\ \wedge$
$month2Nat\ October = 10\ \wedge$
$month2Nat\ November = 11\ \wedge$
$month2Nat\ December = 12$

2 Login

section *login* **parents** *general*

When we go to create a new account, we need some information from the user. We update all of our sets to reflect the new addition. After they have successfully created an account, they are taken to the main menu.

$$
\begin{array}{l}
\underline{\quad CreateAccount \quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad} \\
\Delta HomestayDatabase \\
E? : Email \\
FN? : FirstName \\
LN? : LastName \\
P? : Password \\
\hline
E? \notin Emails \\
Emails' = Emails \cup \{E?\} \\
Passwords' = Passwords \cup \{P?\} \\
Applicants' = Applicants \cup \{(E?, P?, FN?, LN?)\} \\
Valid' = Valid \\
Invalid' = Invalid
\end{array}
$$

To login, a user needs to enter their email and password. At this point they are taken to the main menu.

$$
\begin{array}{l}
\underline{\quad Login \quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad} \\
\Xi HomestayDatabase \\
E? : Email \\
P? : Password \\
FN : FirstName \\
LN : LastName \\
Resp! : Response \\
\hline
E? \in Emails \\
P? \in Passwords \\
(E?, P?, FN, LN) \in Applicants \\
Resp! = LoginSuccessful
\end{array}
$$

Users can reset their password if they forget it. We take an email address, generate a password token, then dish that off to the email address. The user then finds the email with the reset token/link and proceeds to reset their password.

This is supposed to be one more step of indirection so that the user wont have their password reset at random. Of course, if the email address is already compromised, it doesn't make much difference.

```
┌─ ForgotPassword ─────────────────────────────────────
│ ΔHomestayDatabase
│ E? : Email
│ T! : PToken
│ ID : ℕ
│ P : Password
│ FN : FirstName
│ LN : LastName
├──────────────────────────────────────────────────
│ E? ∈ Emails
│ (E?, P, FN, LN) ∈ Applicants
│ ID = # Valid + # Invalid + 1
│ T! = ID ↦ E?
│ Valid' = Valid ∪ {T!}
│ Invalid' = Invalid
└──────────────────────────────────────────────────
```

Once the user has the password token, they can enter their new password. We let them know that the reset was successful.

```
┌─ ResetPassword ──────────────────────────────────────
│ ΔHomestayDatabase
│ P?, P : Password
│ T? : PToken
│ Resp! : Response
│ ID : ℕ
│ E : Email
│ FN : FirstName
│ LN : LastName
│ App_0, App_1 : Applicant
├──────────────────────────────────────────────────
│ T? ∈ Valid
│ E = second  T?
│ Valid' = Valid \ {T?}
│ Invalid' = Invalid ∪ {T?}
│ App_0 = (E, P, FN, LN) ∈ Applicants
│ App_1 = (E, P?, FN, LN)
│ Applicants' = Applicants ⊗ App_1
│ Resp! = PasswordResetSuccessful
└──────────────────────────────────────────────────
```

3   Student

section *student* **parents** *general*

4 Host

The host section allows the user to view and modify preferences specific to hosting a location. There are some general preferences for each host, e.g. smoking, and pets. Then there are preferences for each host location, e.g. price and availability.

$Smoking ::= EnjoysSmoking \mid NonSmoking$
$Pets ::= NoPets \mid YesPets$

$Price == \mathbb{N}$
$Availability == Date \times Date$
$HostPreference == Applicant \times Smoking \times Pets$

EditHostSmoking

$H?, H! : HostPreference$
$S? : Smoking$

$H! = updateSecondOf3(H?, S?)$

EditHostPets

$H?, H! : HostPreference$
$P? : Pets$

$H! = updateThirdOf3(H?, P?)$

$EditHostPreferences == EditHostSmoking \lor EditHostPets$

```
┌─ ValiDate ──────────────────────────────────────────────
│ A? : Availability
│ Resp! : Response
│ Start, End : Date
│ Y_0, Y_1 : Year
│ M_0, M_1 : Month
│ D_0, D_1 : Day
├─────────────────────────────────────────────────────────
│ ((M_0, D_0, Y_0), (M_1, D_1, Y_1)) = A?
│ (Y_0 < Y_1) ∨
│       (Y_0 = Y_1 ∧ month2Nat M_0 < month2Nat M_1) ∨
│       (Y_0 = Y_1 ∧ M_0 = M_1 ∧ D_0 < D_1)
│ Resp! = ValidAvailability
└─────────────────────────────────────────────────────────
```

$((M_0, D_0, Y_0), (M_1, D_1, Y_1)) = A?$

$(Y_0 < Y_1) \vee$
$\quad (Y_0 = Y_1 \wedge month2Nat\ M_0 < month2Nat\ M_1) \vee$
$\quad (Y_0 = Y_1 \wedge M_0 = M_1 \wedge D_0 < D_1)$

$Resp! = ValidAvailability$

5 Administrator

section *admin* **parents** *general*

6 Group Chat

section $group_chat$ **parents** $general$

7 Profile

section *profile* **parents** *general*