

Contents

| | | |
|----------|-------------------------------|-----------|
| 1 | General | 2 |
| 1.1 | Application State | 5 |
| 1.2 | General Operators | 6 |
| 1.2.1 | General Projections | 6 |
| 1.2.2 | General Updates | 7 |
| 1.3 | Morphisms | 11 |
| 1.3.1 | Month | 11 |
| 1.3.2 | InappropriateFlag | 12 |
| 2 | Login | 13 |
| 3 | Student | 17 |
| 4 | Host | 18 |
| 4.1 | Preferences | 18 |
| 4.2 | Validation | 20 |
| 4.3 | Postings | 21 |
| 5 | Administrator | 24 |
| 6 | Group Chat | 27 |
| 7 | Profile | 28 |

1 General

section *general* **parents** *homestay, updates, projections*

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
, StreetNumber
, StreetName
, City
, State
, ZipCode
, Phone
, Photo
]
```

We create a couple of synonyms for photo types.

```
PostingPhoto == Photo
UserPhoto == Photo
```

We're going to construct a date type.

```

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

```

We'll need a couple of flags.

We want to know whether or not the user is an admin or not. This is used so that they get access to certain features in the application.

```

AdminFlag ::=
  Admin
  | NotAdmin
InappropriateFlag ::=
  Zero
  | First
  | Second
  | Third
  | Remove

```

We need to know about the host stuff.

```

Smoking ::=
    EnjoysSmoking
    | NonSmoking
Pets ::=
    NoPets
    | YesPets
Children ::=
    One
    | Two
    | ThreePlus
Diet ::=
    GlutenFree
    | Omnivore
    | Pescatarian
    | Vegetarian
    | Vegan
    | OtherDiet
Religion ::=
    Agnostic
    | Athiest
    | Buddhist
    | Christian
    | Catholic
    | Mormon
    | Muslim
    | OtherReligion
ExactOrArea ::=
    ExactLocation
    | Area

```

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.

```

Applicant == Email × Password × FirstName × LastName × AdminFlag
PToken ==  $\mathbb{N}$  × Email
IFlag == Email × InappropriateFlag

```

Now want to create some stuff specific to the host, but that needs to be used by other parts of the application.

$Price == \mathbb{N}$
 $SizeOfRoom == \mathbb{N} \times \mathbb{N}$
 $Address == StreetNumber \times StreetName \times City \times State \times ZipCode$
 $AddressInfo == Address \times ExactOrArea$
 $Availability == Date \times Date$
 $HostPreference == Applicant \times Smoking \times Pets \times Children \times Diet \times Religion$
 $Posting == HostPreference \times AddressInfo \times SizeOfRoom \times Price \times Availability \times PostingPhoto$

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.

$Response ::=$
 $\quad InvalidToken$
 $\quad | PasswordResetSuccessful$
 $\quad | LoginSuccessful$
 $\quad | InvalidAvailability$
 $\quad | ValidAvailability$

1.1 Application State

Our initial state is just a bunch of empty sets.

$HomestayInitial$

 $Applicants : \mathbb{P} Applicant$
 $Emails : \mathbb{P} Email$
 $Passwords : \mathbb{P} Password$
 $PhoneNumbers : \mathbb{P}(Applicant \times Phone)$
 $UserPhotos : \mathbb{P}(Applicant \times Photo)$
 $Valid, Invalid : \mathbb{P} PToken$
 $IFlags : \mathbb{P} IFlag$
 $Postings : \mathbb{P} Posting$
 $HostPreferences : \mathbb{P} HostPreference$

 $Applicants = \emptyset$
 $Emails = \emptyset$
 $Passwords = \emptyset$
 $PhoneNumbers = \emptyset$
 $UserPhotos = \emptyset$
 $Valid = Invalid = \emptyset$
 $IFlags = \emptyset$
 $Postings = \emptyset$
 $HostPreferences = \emptyset$

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.

| |
|---|
| <i>HomestayDatabase</i> |
| <i>HomestayInitial</i> |
| $\forall e_0, e_1 : \text{Email}; p_0, p_1 : \text{Password}; fn_0, fn_1 : \text{FirstName}; ln_0, ln_1 : \text{LastName} \bullet$ $e_0 = e_1 \Rightarrow (p_0 = p_1 \wedge fn_0 = fn_1 \wedge ln_0 = ln_1)$ $\text{Valid} \cap \text{Invalid} = \emptyset$ |

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 $(_ \otimes _)$

| |
|---|
| $_ \otimes _ : \mathbb{P} \text{Applicant} \times \text{Applicant} \rightarrow \mathbb{P} \text{Applicant}$ |
| $\forall a_1 : \text{Applicant}; as : \mathbb{P} \text{Applicant} \bullet$ $\exists p_0, p_1 : \text{Password};$ $fn_0, fn_1 : \text{FirstName};$ $ln_0, ln_1 : \text{LastName};$ $ad_0, ad_1 : \text{AdminFlag};$ $e : \text{Email};$ $a_0 : \text{Applicant} \mid$ $a_1 = (e, p_1, fn_1, ln_1, ad_1) \wedge (a_0 = (e, p_0, fn_0, ln_0, ad_0) \in as) \bullet$ $as \otimes a_1 = (as \setminus \{a_0\}) \cup \{a_1\}$ |

1.2 General Operators

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

1.2.1 General Projections

section *projections* **parents** *homestay*

| |
|--|
| $[A, B, C]$ |
| $firstOf3 == \lambda a : A; b : B; c : C \bullet a$ $secondOf3 == \lambda a : A; b : B; c : C \bullet b$ $thirdOf3 == \lambda a : A; b : B; c : C \bullet c$ |

$$\begin{array}{l}
\text{firstOf4} == \lambda a : A; b : B; c : C; d : D \bullet a \\
\text{secondOf4} == \lambda a : A; b : B; c : C; d : D \bullet b \\
\text{thirdOf4} == \lambda a : A; b : B; c : C; d : D \bullet c \\
\text{fourthOf4} == \lambda a : A; b : B; c : C; d : D \bullet d
\end{array}$$

$$\begin{array}{l}
\text{firstOf5} == \lambda a : A; b : B; c : C; d : D; e : E \bullet a \\
\text{secondOf5} == \lambda a : A; b : B; c : C; d : D; e : E \bullet b \\
\text{thirdOf5} == \lambda a : A; b : B; c : C; d : D; e : E \bullet c \\
\text{fourthOf5} == \lambda a : A; b : B; c : C; d : D; e : E \bullet d \\
\text{fifthOf5} == \lambda a : A; b : B; c : C; d : D; e : E \bullet e
\end{array}$$

$$\begin{array}{l}
\text{firstOf6} == \lambda a : A; b : B; c : C; d : D; e : E; f : F \bullet a \\
\text{secondOf6} == \lambda a : A; b : B; c : C; d : D; e : E; f : F \bullet b \\
\text{thirdOf6} == \lambda a : A; b : B; c : C; d : D; e : E; f : F \bullet c \\
\text{fourthOf6} == \lambda a : A; b : B; c : C; d : D; e : E; f : F \bullet d \\
\text{fifthOf6} == \lambda a : A; b : B; c : C; d : D; e : E; f : F \bullet e \\
\text{sixthOf6} == \lambda a : A; b : B; c : C; d : D; e : E; f : F \bullet f
\end{array}$$

1.2.2 General Updates

section *updates* **parents** *homestay*

Updating for 3-tuples.

$$\begin{array}{l}
\text{updateFirstOf3} : (A \times B \times C) \times A \rightarrow \\
\quad A \times B \times C \\
\forall abc : A \times B \times C; a : A \bullet \\
\quad \exists a_0 : A; b : B; c : C \mid abc = (a_0, b, c) \bullet \\
\quad \text{updateFirstOf3}(abc, a) = (a, b, c)
\end{array}$$

$$\frac{[A, B, C] \quad \text{updateSecondOf3} : (A \times B \times C) \times B \rightarrow A \times B \times C}{\forall abc : A \times B \times C; b : B \bullet \exists a : A; b_0 : B; c : C \mid abc = (a, b_0, c) \bullet \text{updateSecondOf3}(abc, b) = (a, b, c)}$$

$$\frac{[A, B, C] \quad \text{updateThirdOf3} : (A \times B \times C) \times C \rightarrow A \times B \times C}{\forall abc : A \times B \times C; c : C \bullet \exists a : A; b : B; c_0 : C \mid abc = (a, b, c_0) \bullet \text{updateThirdOf3}(abc, c) = (a, b, c)}$$

Updating for 4-tuples.

$$\begin{array}{c} \overline{[A, B, C, D]} \\ \text{updateFirstOf4} : (A \times B \times C \times D) \times A \rightarrow \\ \quad A \times B \times C \times D \\ \hline \forall abcd : A \times B \times C \times D; a : A \bullet \\ \quad \exists a_0 : A; b : B; c : C; d : D \mid abcd = (a_0, b, c, d) \bullet \\ \quad \text{updateFirstOf4}(abcd, a) = (a, b, c, d) \end{array}$$

$$\begin{array}{c} \text{updateSecondOf4} : (A \times B \times C \times D) \times B \rightarrow \\ \quad A \times B \times C \times D \\ \hline \forall abcd : A \times B \times C \times D; b : B \bullet \\ \quad \exists a : A; b_0 : B; c : C; d : D \mid abcd = (a, b_0, c, d) \bullet \\ \quad \text{updateSecondOf4}(abcd, b) = (a, b, c, d) \end{array}$$

$$\begin{array}{l} \text{updateThirdOf4} : (A \times B \times C \times D) \times C \rightarrow \\ \quad A \times B \times C \times D \\ \hline \forall abcd : A \times B \times C \times D; c : C \bullet \\ \quad \exists a : A; b : B; c_0 : C; d : D \mid abcd = (a, b, c_0, d) \bullet \\ \quad \text{updateThirdOf4}(abcd, c) = (a, b, c, d) \end{array}$$

$$\begin{array}{c}
\text{---} \\
\text{---} [A, B, C, D] \text{---} \\
\text{---} \\
\text{---} \text{updateFourthOf4} : (A \times B \times C \times D) \times D \rightarrow \\
\text{---} \quad \quad \quad A \times B \times C \times D \\
\text{---} \\
\forall abcd : A \times B \times C \times D; d : D \bullet \\
\quad \exists a : A; b : B; c : C; d_0 : D \mid abcd = (a, b, c, d_0) \bullet \\
\quad \text{updateFourthOf4}(abcd, d) = (a, b, c, d)
\end{array}$$

Updating for 5-tuples.

$$\begin{array}{c}
\text{---} \\
\text{---} [A, B, C, D, E] \text{---} \\
\text{---} \\
\text{---} \text{updateFirstOf5} : (A \times B \times C \times D \times E) \times A \rightarrow \\
\text{---} \quad \quad \quad A \times B \times C \times D \times E \\
\text{---} \\
\forall abcde : A \times B \times C \times D \times E; a : A \bullet \\
\quad \exists a_0 : A; b : B; c : C; d : D; e : E \mid abcde = (a_0, b, c, d, e) \bullet \\
\quad \text{updateFirstOf5}(abcde, a) = (a, b, c, d, e)
\end{array}$$

$$\begin{array}{c}
\text{---} \\
\text{---} [A, B, C, D, E] \text{---} \\
\text{---} \\
\text{---} \text{updateSecondOf5} : (A \times B \times C \times D \times E) \times B \rightarrow \\
\text{---} \quad \quad \quad A \times B \times C \times D \times E \\
\text{---} \\
\forall abcde : A \times B \times C \times D \times E; b : B \bullet \\
\quad \exists a : A; b_0 : B; c : C; d : D; e : E \mid abcde = (a, b_0, c, d, e) \bullet \\
\quad \text{updateSecondOf5}(abcde, b) = (a, b, c, d, e)
\end{array}$$

$$\begin{array}{c}
\text{---} \\
\text{---} [A, B, C, D, E] \text{---} \\
\text{---} \\
\text{---} \text{updateThirdOf5} : (A \times B \times C \times D \times E) \times C \rightarrow \\
\text{---} \quad \quad \quad A \times B \times C \times D \times E \\
\text{---} \\
\forall abcde : A \times B \times C \times D \times E; c : C \bullet \\
\quad \exists a : A; b : B; c_0 : C; d : D; e : E \mid abcde = (a, b, c_0, d, e) \bullet \\
\quad \text{updateThirdOf5}(abcde, c) = (a, b, c, d, e)
\end{array}$$

$$\begin{array}{c}
\text{---} \\
\text{---} [A, B, C, D, E] \text{---} \\
\text{---} \\
\text{---} \text{updateFourthOf5} : (A \times B \times C \times D \times E) \times D \rightarrow \\
\text{---} \quad \quad \quad A \times B \times C \times D \times E \\
\text{---} \\
\forall abcde : A \times B \times C \times D \times E; d : D \bullet \\
\quad \exists a : A; b : B; c : C; d_0 : D; e : E \mid abcde = (a, b, c, d_0, e) \bullet \\
\quad \text{updateFourthOf5}(abcde, d) = (a, b, c, d, e)
\end{array}$$

| |
|---|
| $[A, B, C, D, E] \equiv \frac{\text{updateFifthOf5} : (A \times B \times C \times D \times E) \times E \rightarrow A \times B \times C \times D \times E}{\forall abcde : A \times B \times C \times D \times E; e : E \bullet \exists a : A; b : B; c : C; d : D; e_0 : E \mid abcde = (a, b, c, d, e_0) \bullet \text{updateFifthOf5}(abcde, e) = (a, b, c, d, e)}$ |
|---|

Updating for 6-tuples.

$$\begin{array}{l} \text{updateFirstOf6} : (A \times B \times C \times D \times E \times F) \times A \rightarrow \\ \quad A \times B \times C \times D \times E \times F \\ \hline \forall abcdef : A \times B \times C \times D \times E \times F; a : A \bullet \\ \quad \exists a_0 : A; b : B; c : C; d : D; e : E; f : F \mid abcdef = (a_0, b, c, d, e, f) \bullet \\ \quad \text{updateFirstOf6}(abcdef, a) = (a, b, c, d, e, f) \end{array}$$

| |
|--|
| $[A, B, C, D, E, F] \equiv \frac{\text{updateSecondOf6} : (A \times B \times C \times D \times E \times F) \times B \rightarrow A \times B \times C \times D \times E \times F}{\begin{array}{l} \forall abcdef : A \times B \times C \times D \times E \times F; b : B \bullet \\ \exists a : A; b_0 : B; c : C; d : D; e : E; f : F \mid abcdef = (a, b_0, c, d, e, f) \bullet \\ \text{updateSecondOf6}(abcdef, b) = (a, b, c, d, e, f) \end{array}}$ |
|--|

$$\begin{array}{l} \text{updateThirdOf6} : (A \times B \times C \times D \times E \times F) \times C \rightarrow \\ \quad A \times B \times C \times D \times E \times F \\ \hline \forall abcdef : A \times B \times C \times D \times E \times F; c : C \bullet \\ \quad \exists a : A; b : B; c_0 : C; d : D; e : E; f : F \mid abcdef = (a, b, c_0, d, e, f) \bullet \\ \quad \text{updateThirdOf6}(abcdef, c) = (a, b, c, d, e, f) \end{array}$$

$$\begin{array}{c} \text{updateFourthOf6} : (A \times B \times C \times D \times E \times F) \times D \rightarrow \\ \quad A \times B \times C \times D \times E \times F \\ \hline \forall abcdef : A \times B \times C \times D \times E \times F; d : D \bullet \\ \quad \exists a : A; b : B; c : C; d_0 : D; e : E; f : F \mid abcdef = (a, b, c, d_0, e, f) \bullet \\ \quad \text{updateFourthOf6}(abcdef, d) = (a, b, c, d, e, f) \end{array}$$

$$\begin{array}{|l}
\hline
\hline
[A, B, C, D, E, F] \text{ } \hline
\hline
\hline
\text{updateFifthOf6} : (A \times B \times C \times D \times E \times F) \times E \rightarrow \\
\qquad\qquad\qquad A \times B \times C \times D \times E \times F \\
\hline
\forall abcdef : A \times B \times C \times D \times E \times F; e : E \bullet \\
\qquad\qquad\qquad \exists a : A; b : B; c : C; d : D; e_0 : E; f : F \mid abcdef = (a, b, c, d, e_0, f) \bullet \\
\qquad\qquad\qquad \text{updateFifthOf6}(abcdef, e) = (a, b, c, d, e, f) \\
\hline
\end{array}$$

$$\begin{array}{|l}
\hline
\hline
[A, B, C, D, E, F] \text{ } \hline
\hline
\hline
\text{updateSixthOf6} : (A \times B \times C \times D \times E \times F) \times F \rightarrow \\
\qquad\qquad\qquad A \times B \times C \times D \times E \times F \\
\hline
\forall abcdef : A \times B \times C \times D \times E \times F; f : F \bullet \\
\qquad\qquad\qquad \exists a : A; b : B; c : C; d : D; e : E; f_0 : F \mid abcdef = (a, b, c, d, e, f_0) \bullet \\
\qquad\qquad\qquad \text{updateSixthOf6}(abcdef, f) = (a, b, c, d, e, f) \\
\hline
\end{array}$$

1.3 Morphisms

1.3.1 Month

$$\begin{array}{|l}
\hline
\text{month2Nat} : \text{Month} \rightarrow \mathbb{N} \\
\hline
\text{month2Nat January} = 1 \wedge \\
\text{month2Nat February} = 2 \wedge \\
\text{month2Nat March} = 3 \wedge \\
\text{month2Nat April} = 4 \wedge \\
\text{month2Nat May} = 5 \wedge \\
\text{month2Nat June} = 6 \wedge \\
\text{month2Nat July} = 7 \wedge \\
\text{month2Nat August} = 8 \wedge \\
\text{month2Nat September} = 9 \wedge \\
\text{month2Nat October} = 10 \wedge \\
\text{month2Nat November} = 11 \wedge \\
\text{month2Nat December} = 12 \\
\hline
\end{array}$$

| |
|--|
| $nat2Month : \mathbb{N} \leftrightarrow Month$ |
| $nat2Month\ 1 = January \wedge$ $nat2Month\ 2 = February \wedge$ $nat2Month\ 3 = March \wedge$ $nat2Month\ 4 = April \wedge$ $nat2Month\ 5 = May \wedge$ $nat2Month\ 6 = June \wedge$ $nat2Month\ 7 = July \wedge$ $nat2Month\ 8 = August \wedge$ $nat2Month\ 9 = September \wedge$ $nat2Month\ 10 = October \wedge$ $nat2Month\ 11 = November \wedge$ $nat2Month\ 12 = December$ |

1.3.2 InappropriateFlag

| |
|--|
| $InFlag2Nat : InappropriateFlag \rightarrow \mathbb{N}$ |
| $InFlag2Nat\ Zero = 0 \wedge$ $InFlag2Nat\ First = 1 \wedge$ $InFlag2Nat\ Second = 2 \wedge$ $InFlag2Nat\ Third = 3 \wedge$ $InFlag2Nat\ Remove = 4$ |

| |
|--|
| $Nat2InFlag : \mathbb{N} \leftrightarrow InappropriateFlag$ |
| $Nat2InFlag\ 0 = Zero \wedge$ $Nat2InFlag\ 1 = First \wedge$ $Nat2InFlag\ 2 = Second \wedge$ $Nat2InFlag\ 3 = Third \wedge$ $Nat2InFlag\ 4 = Remove$ |

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.

CreateUserAccount

Δ *HomestayDatabase*

$E? : Email$

$FN? : FirstName$

$LN? : LastName$

$P? : Password$

$E? \notin Emails$

$Emails' = Emails \cup \{E?\}$

$Passwords' = Passwords \cup \{P?\}$

$Applicants' = Applicants \cup \{(E?, P?, FN?, LN?, NotAdmin)\}$

$PhoneNumbers' = PhoneNumbers$

$UserPhoto' = UserPhoto$

$Valid' = Valid$

$Invalid' = Invalid$

$Postings' = Postings$

$HostPreferences' = HostPreferences$

| |
|---|
| $CreateAdminAccount$ $\Delta HomestayDatabase$ $E? : Email$ $FN? : FirstName$ $LN? : LastName$ $P? : Password$ $AD? : AdminFlag$ |
| $E? \notin Emails$ $AD? = Admin$ $Emails' = Emails \cup \{E?\}$ $Passwords' = Passwords \cup \{P?\}$ $Applicants' = Applicants \cup \{(E?, P?, FN?, LN?, AD?)\}$ $PhoneNumbers' = PhoneNumbers$ $UserPhoto' = UserPhoto$ $Valid' = Valid$ $Invalid' = Invalid$ $Postings' = Posting$ $HostPreferences' = HostPreferences$ |

A person can create either a user account, or they can create an admin account. The idea is that the admin account isn't something that you can specify, but you must be given a link to sign up for. The link you follow gives the admin flag.

$$CreateAccount == CreateUserAccount \vee CreateAdminAccount$$

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

| |
|---|
| $Login$ $\Xi HomestayDatabase$ $E? : Email$ $P? : Password$ $FN : FirstName$ $LN : LastName$ $AD : AdminFlag$ $Resp! : Response$ |
| $E? \in Emails$ $P? \in Passwords$ $(E?, P?, FN, LN, AD) \in Applicants$ $Resp! = LoginSuccessful$ |

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 won't have their password reset at random. Of course, if the email address is already compromised, it doesn't make much difference.

| | |
|--------------------------------------|--|
| <i>ForgotPassword</i> | |
| $\Delta HomestayDatabase$ | |
| $E? : Email$ | |
| $T! : PToken$ | |
| $ID : \mathbb{N}$ | |
| $P : Password$ | |
| $FN : FirstName$ | |
| $LN : LastName$ | |
| $AD : AdminFlag$ | |
| $E? \in Emails$ | |
| $(E?, P, FN, LN, AD) \in Applicants$ | |
| $ID = \# Valid + \# Invalid + 1$ | |
| $T! = ID \mapsto E?$ | |
| $Valid' = Valid \cup \{T!\}$ | |
| $Invalid' = Invalid$ | |
| $Applicants' = Applicants$ | |
| $Emails' = Emails$ | |
| $Passwords' = Passwords$ | |
| $PhoneNumbers' = PhoneNumbers$ | |
| $UserPhoto' = UserPhoto$ | |
| $Postings' = Postings$ | |
| $HostPreferences' = HostPreferences$ | |

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

ResetPassword

$\Delta HomestayDatabase$

$P?, P : Password$

$T? : PToken$

$Resp! : Response$

$ID : \mathbb{N}$

$E : Email$

$FN : FirstName$

$LN : LastName$

$AD : AdminFlag$

$App_0, App_1 : Applicant$

$T? \in Valid$

$E = second\ T?$

$Valid' = Valid \setminus \{T?\}$

$Invalid' = Invalid \cup \{T?\}$

$App_0 = (E, P, FN, LN, AD) \in Applicants$

$App_1 = (E, P?, FN, LN, AD)$

$Applicants' = Applicants \otimes App_1$

$Resp! = PasswordResetSuccessful$

$Emails' = Emails$

$Passwords' = Passwords$

$PhoneNumbers' = PhoneNumbers$

$Postings' = Postings$

$HostPreferences' = HostPreferences$

3 Student

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

4 Host

section *host* parents *general*

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.

4.1 Preferences

| |
|--|
| $updateHostPrefs : \mathbb{P} HostPreference \times HostPreference \rightarrow \mathbb{P} HostPreference$ |
| $\forall hp : HostPreference; hps : \mathbb{P} HostPreference \bullet$ $\exists hp_0 : HostPreference \mid$ $firstOf6 hp_0 = firstOf6 hp \bullet$ $updateHostPrefs(hps, hp) = (hps \setminus \{hp_0\}) \cup \{hp\}$ |

| |
|---|
| <i>EditHostSmoking</i> |
| $\Delta HomestayDatabase$ $H? : HostPreference$ $H : HostPreference$ $S? : Smoking$ |
| $H? \in HostPreferences$ $H = updateSecondOf6(H?, S?)$ $HostPreferences' = updateHostPrefs(HostPreferences, H)$ $Applicants' = Applicants$ $Emails' = Emails$ $Passwords' = Passwords$ $PhoneNumbers' = PhoneNumbers$ $UserPhotos' = UserPhotos$ $Valid' = Valid$ $Invalid' = Invalid$ $Postings' = Postings$ |

EditHostPets

Δ HomestayDatabase

$H?, H : \text{HostPreference}$

$P? : \text{Pets}$

$H? \in \text{HostPreferences}$

$H = \text{updateThirdOf6}(H?, P?)$

$\text{HostPreferences}' = \text{updateHostPrefs}(\text{HostPreferences}, H)$

$\text{Applicants}' = \text{Applicants}$

$\text{Emails}' = \text{Emails}$

$\text{Passwords}' = \text{Passwords}$

$\text{PhoneNumbers}' = \text{PhoneNumbers}$

$\text{UserPhotos}' = \text{UserPhotos}$

$\text{Valid}' = \text{Valid}$

$\text{Invalid}' = \text{Invalid}$

$\text{Postings}' = \text{Postings}$

EditHostChildren

Δ HomestayDatabase

$H?, H : \text{HostPreference}$

$C? : \text{Children}$

$H? \in \text{HostPreferences}$

$H = \text{updateFourthOf6}(H?, C?)$

$\text{HostPreferences}' = \text{updateHostPrefs}(\text{HostPreferences}, H)$

$\text{Applicants}' = \text{Applicants}$

$\text{Emails}' = \text{Emails}$

$\text{Passwords}' = \text{Passwords}$

$\text{PhoneNumbers}' = \text{PhoneNumbers}$

$\text{UserPhotos}' = \text{UserPhotos}$

$\text{Valid}' = \text{Valid}$

$\text{Invalid}' = \text{Invalid}$

$\text{Postings}' = \text{Postings}$

EditHostDiet

Δ HomestayDatabase

$H?, H : \text{HostPreference}$

$D? : \text{Diet}$

$H? \in \text{HostPreferences}$

$H = \text{updateFifthOf6}(H?, D?)$

$\text{HostPreferences}' = \text{updateHostPrefs}(\text{HostPreferences}, H)$

$\text{Applicants}' = \text{Applicants}$

$\text{Emails}' = \text{Emails}$

$\text{Passwords}' = \text{Passwords}$

$\text{PhoneNumbers}' = \text{PhoneNumbers}$

$\text{UserPhotos}' = \text{UserPhotos}$

$\text{Valid}' = \text{Valid}$

$\text{Invalid}' = \text{Invalid}$

$\text{Postings}' = \text{Postings}$

EditHostReligion

Δ HomestayDatabase

$H?, H : \text{HostPreference}$

$R? : \text{Religion}$

$H? \in \text{HostPreferences}$

$H = \text{updateSixthOf6}(H?, R?)$

$\text{HostPreferences}' = \text{updateHostPrefs}(\text{HostPreferences}, H)$

$\text{Applicants}' = \text{Applicants}$

$\text{Emails}' = \text{Emails}$

$\text{Passwords}' = \text{Passwords}$

$\text{PhoneNumbers}' = \text{PhoneNumbers}$

$\text{UserPhotos}' = \text{UserPhotos}$

$\text{Valid}' = \text{Valid}$

$\text{Invalid}' = \text{Invalid}$

$\text{Postings}' = \text{Postings}$

$\text{EditHostPreferences} == \text{EditHostSmoking} \vee$

$\text{EditHostPets} \vee$

$\text{EditHostChildren} \vee$

$\text{EditHostDiet} \vee$

EditHostReligion

4.2 Validation

So long as the start date is before the end date, we consider the Availability to be valid.

| |
|--|
| <i>ValiDateGood</i> |
| <i>Avail?</i> : <i>Availability</i> <i>AvailResp!</i> : <i>Response</i> <i>Start, End</i> : <i>Date</i> <i>Y₀, Y₁</i> : <i>Year</i> <i>M₀, M₁</i> : <i>Month</i> <i>D₀, D₁</i> : <i>Day</i> |
| $((M_0, D_0, Y_0), (M_1, D_1, Y_1)) = Avail?$ $(Y_0 < Y_1) \vee$ $(Y_0 = Y_1 \wedge month2Nat\ M_0 < month2Nat\ M_1) \vee$ $(Y_0 = Y_1 \wedge M_0 = M_1 \wedge D_0 < D_1)$ <i>AvailResp!</i> = <i>ValidAvailability</i> |

If the start date is after the end date, then no bueno.

| |
|--|
| <i>ValiDateBad</i> |
| <i>Avail?</i> : <i>Availability</i> <i>AvailResp!</i> : <i>Response</i> <i>Start, End</i> : <i>Date</i> <i>Y₀, Y₁</i> : <i>Year</i> <i>M₀, M₁</i> : <i>Month</i> <i>D₀, D₁</i> : <i>Day</i> |
| $((M_0, D_0, Y_0), (M_1, D_1, Y_1)) = Avail?$ $(Y_0 > Y_1) \vee$ $(Y_0 = Y_1 \wedge month2Nat\ M_0 > month2Nat\ M_1) \vee$ $(Y_0 = Y_1 \wedge M_0 = M_1 \wedge D_0 > D_1)$ <i>AvailResp!</i> = <i>InvalidAvailability</i> |

$$ValiDate == ValiDateGood \vee ValiDateBad$$

4.3 Postings

To create a new posting, the host enters all the relevant information for hosting

NewPosting

Δ *HomestayDatabase*

ValiDate

$H? : \textit{HostPreference}$

$\textit{Addr}? : \textit{Address}$

$E? : \textit{ExactOrArea}$

$\textit{WIDTH}? : \mathbb{N}$

$\textit{LENGTH}? : \mathbb{N}$

$\textit{PR}? : \textit{Price}$

$\textit{Avail}? : \textit{Availability}$

$\textit{PH}? : \textit{Photo}$

$P : \textit{Posting}$

$\textit{AI} : \textit{AddressInfo}$

$S : \textit{SizeOfRoom}$

$\textit{AvailResp}! : \textit{Response}$

$\textit{AvailResp}! = \textit{ValidAvailability}$

$\textit{AI} = \textit{Addr}? \mapsto E?$

$S = \textit{WIDTH}? \mapsto \textit{LENGTH}?$

$P = (H?, \textit{AI}, S, \textit{PR}?, \textit{Avail}?, \textit{PH}?)$

$\textit{Postings}' = \textit{Postings} \cup \{P\}$

$\textit{Applicants}' = \textit{Applicants}$

$\textit{Emails}' = \textit{Emails}$

$\textit{Passwords}' = \textit{Passwords}$

$\textit{PhoneNumbers}' = \textit{PhoneNumbers}$

$\textit{UserPhotos}' = \textit{UserPhotos}$

$\textit{Valid}' = \textit{Valid}$

$\textit{Invalid}' = \textit{Invalid}$

$\textit{HostPreferences}' = \textit{HostPreferences}$

Deleting a post removes the post from Postings

DeletePosting

$\Delta HomestayDatabase$

$P? : Posting$

$P? \in Postings$

$Postings' = Postings \setminus \{P?\}$

$Applicants' = Applicants$

$Emails' = Emails$

$Passwords' = Passwords$

$PhoneNumbers' = PhoneNumbers$

$UserPhotos' = UserPhotos$

$Valid' = Valid$

$Invalid' = Invalid$

$HostPreferences' = HostPreferences$

5 Administrator

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

Administrator login only needs to act like an inbox with "flag as inappropriate" or "accept/reject match" messages from students. The administrator will have a link to the group chat that is flagged and can add messages or end the chat if it is inappropriate. If the student and host decide to chat, once they send in their accept/reject match to the admin, the admin will have final approval to accept or reject the match.

$$\begin{aligned} MatchFlag ::= & \\ & Accept \\ & | Reject \end{aligned}$$

$$\begin{aligned} Message ::= & \\ & OneAccept \\ & | TwoAccepts \\ & | NoAccepts \\ & | Inappropriate \end{aligned}$$

| | |
|--|--|
| $AdminMatch : Message \rightarrow MatchFlag$ | $AdminMatch \ OneAccept = Reject \wedge$ $AdminMatch \ NoAccepts = Reject \wedge$ $AdminMatch \ TwoAccepts = Accept$ |
|--|--|

When we need to get rid of a user, we have some steps to take

- Remove the applicant.
- Remove the email corresponding to the applicant.
- Remove the password corresponding to the applicant.

RemoveUser

Δ *HomestayDatabase*

$A? : \text{Applicant}$

$E : \text{Email}$

$P : \text{Password}$

$E = \text{firstOf5 } A?$

$P = \text{secondOf5 } A?$

$\text{Applicants}' = \text{Applicants} \setminus \{A?\}$

$\text{Emails}' = \text{Emails} \setminus \{E\}$

$\text{Passwords}' = \text{Passwords} \setminus \{P\}$

$\text{HostPreferences}' = \text{HostPreferences}$

$\text{PhoneNumbers}' = \text{PhoneNumbers}$

$\text{UserPhotos}' = \text{UserPhotos}$

$\text{Valid}' = \text{Valid}$

$\text{Invalid}' = \text{Invalid}$

$\text{Postings}' = \text{Postings}$

AdminInappropriate

Δ *HomestayDatabase*

$M? : \text{Message}$

$E? : \text{Email}$

$I : \text{IFlag}$

$In, In2 : \text{InappropriateFlag}$

$M? = \text{Inappropriate}$

$\{E? \mapsto In\} = \{E?\} \triangleleft \text{IFlags}$

$In \neq \text{Remove}$

$In2 = \text{Nat2InFlag}((\text{InFlag2Nat } In) + 1)$

$\text{IFlags}' = \text{IFlags} \oplus \{E? \mapsto In2\}$

$In2 \neq \text{Remove}$

$\text{Applicants}' = \text{Applicants}$

$\text{Emails}' = \text{Emails}$

$\text{Passwords}' = \text{Passwords}$

$\text{PhoneNumbers}' = \text{PhoneNumbers}$

$\text{UserPhotos}' = \text{UserPhotos}$

$\text{Valid}' = \text{Valid}$

$\text{Invalid}' = \text{Invalid}$

InappropriateUser _____

$\Delta HomestayDatabase$

RemoveUser

$M? : Message$

$E? : Email$

$I : IFlag$

$In, In2 : InappropriateFlag$

$M? = Inappropriate$

$\{E? \mapsto In\} = \{E?\} \triangleleft IFlags$

$In \neq Remove$

$In2 = Nat2InFlag((InFlag2Nat\ In) + 1)$

$IFlags' = IFlags \oplus \{E? \mapsto In2\}$

$In2 = Remove$

6 Group Chat

section *group_{chat}* **parents** *general*

7 Profile

section *profile* parents *general*

The profile section has functionality to update/change basic user information: First Name, Last Name, User Photo, Email, and Phone Number.

We just need to replace the first name for the applicant.

| |
|--|
| <i>EditFirstName</i> |
| $\Delta HomestayDatabase$ |
| $FN? : FirstName$ |
| $App_0?, App_1 : Applicant$ |
| $App_0? \in Applicants$ |
| $App_1 = updateThirdOf5(App_0?, FN?)$ |
| $Applicants' = Applicants \otimes App_1$ |
| $Emails' = Emails$ |
| $Passwords' = Passwords$ |
| $PhoneNumbers' = PhoneNumbers$ |
| $Valid' = Valid$ |
| $Invalid' = Invalid$ |
| $HostPreferences' = HostPreferences$ |
| $Postings' = Postings$ |
| $UserPhotos' = UserPhotos$ |

We just need to replace the last name for the applicant.

| |
|--|
| <i>EditLastName</i> |
| $\Delta HomestayDatabase$ |
| $LN? : LastName$ |
| $App_0?, App_1 : Applicant$ |
| $App_0? \in Applicants$ |
| $App_1 = updateFourthOf5(App_0?, LN?)$ |
| $Applicants' = Applicants \otimes App_1$ |
| $Emails' = Emails$ |
| $Passwords' = Passwords$ |
| $PhoneNumbers' = PhoneNumbers$ |
| $UserPhotos' = UserPhotos$ |
| $Valid' = Valid$ |
| $Invalid' = Invalid$ |
| $HostPreferences' = HostPreferences$ |
| $Postings' = Postings$ |

We just need to replace the photo for the applicant.

| |
|--|
| <i>EditPhoto</i> |
| $\Delta HomestayDatabase$ |
| $UP? : UserPhoto$ |
| $App? : Applicant$ |
| $UserPhotos' = UserPhotos \oplus \{App? \mapsto UP?\}$ |
| $Applicants' = Applicants$ |
| $Emails' = Emails$ |
| $Passwords' = Passwords$ |
| $PhoneNumbers' = PhoneNumbers$ |
| $Valid' = Valid$ |
| $Invalid' = Invalid$ |
| $HostPreferences' = HostPreferences$ |
| $Postings' = Postings$ |

We have to check that the email is not currently used in the system. Then update the email for the applicant and ensure the changes cascade throughout the system.

| |
|---|
| <i>EditEmail</i> |
| $\Delta HomestayDatabase$ |
| $FN : FirstName$ |
| $LN : LastName$ |
| $P : Password$ |
| $E?, E : Email$ |
| $AD : AdminFlag$ |
| $\exists App_0, App_1 : Applicant \mid App_0 = (E, P, FN, LN, AD) \in Applicants \wedge$ $App_1 = (E?, P, FN, LN, AD) \notin Applicants \bullet$ |
| $Applicants' = Applicants \setminus \{App_0\} \wedge Applicants' = Applicants \cup \{App_1\}$ |
| $Emails' = Emails$ |
| $Passwords' = Passwords$ |
| $PhoneNumbers' = PhoneNumbers$ |
| $UserPhotos' = UserPhotos$ |
| $Valid' = Valid$ |
| $Invalid' = Invalid$ |
| $HostPreferences' = HostPreferences$ |
| $Postings' = Postings$ |

We just need to update the phone number for the applicant.

EditPhone

$\Delta HomestayDatabase$

$PNUM? : Phone$

$App? : Applicant$

$PhoneNumbers' = PhoneNumbers \oplus \{App? \mapsto PNUM?\}$

$Applicants' = Applicants$

$Emails' = Emails$

$Passwords' = Passwords$

$UserPhotos' = UserPhotos$

$Valid' = Valid$

$Invalid' = Invalid$

$HostPreferences' = HostPreferences$

$Postings' = Postings$