

## Contents

<b>1</b>	<b>General</b>	<b>2</b>
1.1	General Operators . . . . .	4
<b>2</b>	<b>Login</b>	<b>7</b>
<b>3</b>	<b>Student</b>	<b>11</b>
<b>4</b>	<b>Host</b>	<b>12</b>
<b>5</b>	<b>Administrator</b>	<b>14</b>
<b>6</b>	<b>Group Chat</b>	<b>15</b>
<b>7</b>	<b>Profile</b>	<b>16</b>

# 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

$[Email, Password, FirstName, LastName]$

$Month ::= January$

| *February*

| *March*

| *April*

| *May*

| *June*

| *July*

| *August*

| *September*

| *October*

| *November*

| *December*

$Day == \mathbb{N}$

$Year == \mathbb{N}$

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*

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.

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

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.

$$\begin{aligned} \textit{Response} ::= & \\ & \textit{InvalidToken} \\ & | \textit{PasswordResetSuccessful} \\ & | \textit{LoginSuccessful} \\ & | \textit{InvalidAvailability} \\ & | \textit{ValidAvailability} \end{aligned}$$

We need to know about the host stuff.

$$\begin{aligned} \textit{Smoking} ::= & \textit{EnjoysSmoking} \mid \textit{NonSmoking} \\ \textit{Pets} ::= & \textit{NoPets} \mid \textit{YesPets} \end{aligned}$$

$$\begin{aligned} \textit{Price} &== \mathbb{N} \\ \textit{Availability} &== \textit{Date} \times \textit{Date} \\ \textit{HostPreference} &== \textit{Applicant} \times \textit{Smoking} \times \textit{Pets} \\ \textit{Posting} &== \textit{HostPreference} \times \textit{Availability} \times \textit{Price} \end{aligned}$$

Our initial state is just a bunch of empty sets.

$\begin{aligned} &\textit{HomestayInitial} \\ &\textit{Applicants} : \mathbb{P} \textit{Applicant} \\ &\textit{Emails} : \mathbb{P} \textit{Email} \\ &\textit{Passwords} : \mathbb{P} \textit{Password} \\ &\textit{Valid}, \textit{Invalid} : \mathbb{P} \textit{PToken} \\ &\textit{Postings} : \mathbb{P} \textit{Posting} \\ &\textit{HostPreferences} : \mathbb{P} \textit{HostPreference} \end{aligned}$
$\begin{aligned} &\textit{Applicants} = \emptyset \\ &\textit{Emails} = \emptyset \\ &\textit{Passwords} = \emptyset \\ &\textit{Valid} = \textit{Invalid} = \emptyset \\ &\textit{Postings} = \emptyset \\ &\textit{HostPreferences} = \emptyset \end{aligned}$

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

## 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.

$[X, Y, Z]$
$\text{firstOf3} == \lambda x : X; y : Y; z : Z \bullet x$ $\text{secondOf3} == \lambda x : X; y : Y; z : Z \bullet y$ $\text{thirdOf3} == \lambda x : X; y : Y; z : Z \bullet z$

$[X, Y, Z]$
$\text{updateFirstOf3} : (X \times Y \times Z) \times X \rightarrow X \times Y \times Z$ $\forall xyz : X \times Y \times Z; x : X \bullet$ $\exists x_0 : X; y : Y; z : Z \mid xyz = (x_0, y, z) \bullet$ $\text{updateFirstOf3}(xyz, x) = (x, y, z)$

$[X, Y, Z]$
$\text{updateSecondOf3} : (X \times Y \times Z) \times Y \rightarrow X \times Y \times Z$ $\forall xyz : X \times Y \times Z; y : Y \bullet$ $\exists x : X; y_0 : Y; z : Z \mid xyz = (x, y_0, z) \bullet$ $\text{updateSecondOf3}(xyz, y) = (x, y, z)$

$[X, Y, Z]$	$updateThirdOf3 : (X \times Y \times Z) \times Z \rightarrow X \times Y \times Z$
$\forall xyz : X \times Y \times Z; z : Z \bullet$	$\exists x : X; y : Y; z_0 : Z \mid xyz = (x, y, z_0) \bullet$
	$updateThirdOf3(xyz, z) = (x, y, z)$

We're going to Greenspun up some stuff.

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

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

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

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

$updateHostPrefs : \mathbb{P} HostPreference \times HostPreference \rightarrow \mathbb{P} HostPreference$
$\forall hp : HostPreference; hps : \mathbb{P} HostPreference \bullet$ $\exists hp_0 : HostPreference \mid$ $firstOf3 hp_0 = firstOf3 hp \bullet$ $updateHostPrefs(hps, hp) = (hps \setminus \{hp_0\}) \cup \{hp\}$

<i>RemoveUser</i>
$\Delta HomestayDatabase$ $A? : Applicant$ $E : Email$ $P : Password$ $FN : FirstName$ $LN : LastName$ $AF : AdminFlag$
$(E, P, FN, LN, AF) = A?$ $Applicants' = Applicants \setminus \{A?\}$ $Emails' = Emails \setminus \{E\}$ $HostPreferences' = HostPreferences$ $Passwords' = Passwords$ $Valid' = Valid$ $Invalid' = Invalid$ $Postings' = Postings$

## 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.

<i>CreateUserAccount</i>
$\Delta HomestayDatabase$
$E? : Email$
$FN? : FirstName$
$LN? : LastName$
$P? : Password$
$E? \notin Emails$
$Emails' = Emails \cup \{E?\}$
$Passwords' = Passwords \cup \{P?\}$
$Applicants' = Applicants \cup \{(E?, P?, FN?, LN?, NotAdmin)\}$
$Valid' = Valid$
$Invalid' = Invalid$
$Postings' = Postings$
$HostPreferences' = HostPreferences$

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

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.

$$\textit{CreateAccount} == \textit{CreateUserAccount} \vee \textit{CreateAdminAccount}$$

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

<i>Login</i>	
$\exists \textit{HomestayDatabase}$	
$E? : \textit{Email}$	
$P? : \textit{Password}$	
$FN : \textit{FirstName}$	
$LN : \textit{LastName}$	
$AD : \textit{AdminFlag}$	
$\textit{Resp!} : \textit{Response}$	
$E? \in \textit{Emails}$	
$P? \in \textit{Passwords}$	
$(E?, P?, FN, LN, AD) \in \textit{Applicants}$	
$\textit{Resp!} = \textit{LoginSuccessful}$	

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



*ForgotPassword*

$\Delta$ *HomestayDatabase*

$E? : \textit{Email}$

$T! : \textit{PToken}$

$ID : \mathbb{N}$

$P : \textit{Password}$

$FN : \textit{FirstName}$

$LN : \textit{LastName}$

$AD : \textit{AdminFlag}$

$E? \in \textit{Emails}$

$(E?, P, FN, LN, AD) \in \textit{Applicants}$

$ID = \# \textit{Valid} + \# \textit{Invalid} + 1$

$T! = ID \mapsto E?$

$\textit{Valid}' = \textit{Valid} \cup \{T!\}$

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

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

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

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

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

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

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

*ResetPassword*

$\Delta \text{HomestayDatabase}$

$P?, P : \text{Password}$

$T? : \text{PToken}$

$\text{Resp!} : \text{Response}$

$ID : \mathbb{N}$

$E : \text{Email}$

$FN : \text{FirstName}$

$LN : \text{LastName}$

$AD : \text{AdminFlag}$

$\text{App}_0, \text{App}_1 : \text{Applicant}$

$T? \in \text{Valid}$

$E = \text{second } T?$

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

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

$\text{App}_0 = (E, P, FN, LN, AD) \in \text{Applicants}$

$\text{App}_1 = (E, P?, FN, LN, AD)$

$\text{Applicants}' = \text{Applicants} \otimes \text{App}_1$

$\text{Resp!} = \text{PasswordResetSuccessful}$

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

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

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

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

### 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.

*EditHostSmoking*

$\Delta HomestayDatabase$

$H? : HostPreference$

$H : HostPreference$

$S? : Smoking$

$H? \in HostPreferences$

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

$HostPreferences' = updateHostPrefs(HostPreferences, H)$

$Applicants' = Applicants$

$Emails' = Emails$

$Passwords' = Passwords$

$Valid' = Valid$

$Invalid' = Invalid$

$Postings' = Postings$

*EditHostPets*

$\Delta HomestayDatabase$

$H?, H : HostPreference$

$P? : Pets$

$H? \in HostPreferences$

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

$HostPreferences' = updateHostPrefs(HostPreferences, H)$

$Applicants' = Applicants$

$Emails' = Emails$

$Passwords' = Passwords$

$Valid' = Valid$

$Invalid' = Invalid$

$Postings' = Postings$

$EditHostPreferences == EditHostSmoking \vee EditHostPets$

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

<i>ValiDateGood</i>
$A? : \textit{Availability}$ $\textit{Resp!} : \textit{Response}$ $\textit{Start}, \textit{End} : \textit{Date}$ $Y_0, Y_1 : \textit{Year}$ $M_0, M_1 : \textit{Month}$ $D_0, D_1 : \textit{Day}$
$((M_0, D_0, Y_0), (M_1, D_1, Y_1)) = A?$ $(Y_0 < Y_1) \vee$ $(Y_0 = Y_1 \wedge \textit{month2Nat } M_0 < \textit{month2Nat } M_1) \vee$ $(Y_0 = Y_1 \wedge M_0 = M_1 \wedge D_0 < D_1)$ $\textit{Resp!} = \textit{ValidAvailability}$

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

<i>ValiDateBad</i>
$A? : \textit{Availability}$ $\textit{Resp!} : \textit{Response}$ $\textit{Start}, \textit{End} : \textit{Date}$ $Y_0, Y_1 : \textit{Year}$ $M_0, M_1 : \textit{Month}$ $D_0, D_1 : \textit{Day}$
$((M_0, D_0, Y_0), (M_1, D_1, Y_1)) = A?$ $(Y_0 > Y_1) \vee$ $(Y_0 = Y_1 \wedge \textit{month2Nat } M_0 > \textit{month2Nat } M_1) \vee$ $(Y_0 = Y_1 \wedge M_0 = M_1 \wedge D_0 > D_1)$ $\textit{Resp!} = \textit{InvalidAvailability}$

$$\textit{ValiDate} == \textit{ValiDateGood} \vee \textit{ValiDateBad}$$

5 Administrator

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

<i>Requests</i>	
<i>PREDs</i>	

## 6 Group Chat

**section** *group<sub>chat</sub>* **parents** *general*

## 7 Profile

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