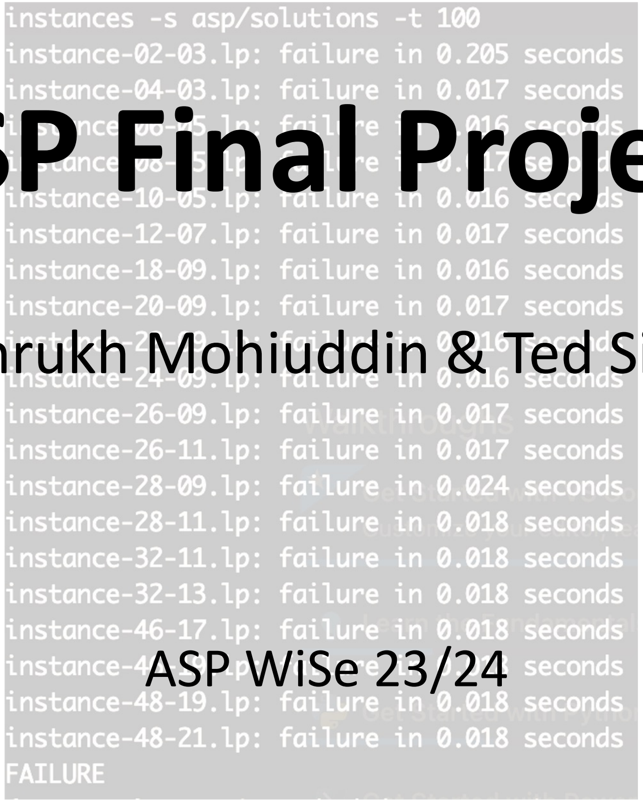
ASP Final Project

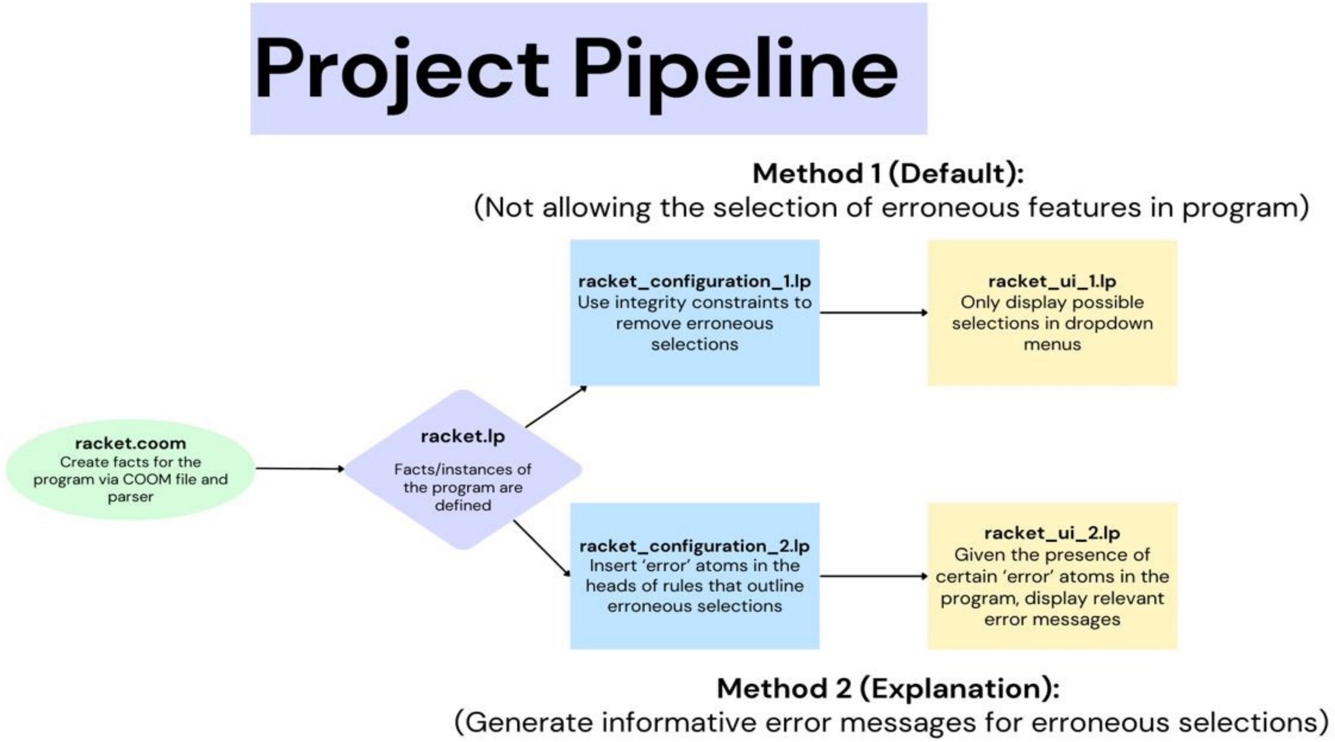
Shahrukh Mohiuddin & Ted Sither

ASP WiSe 23/24



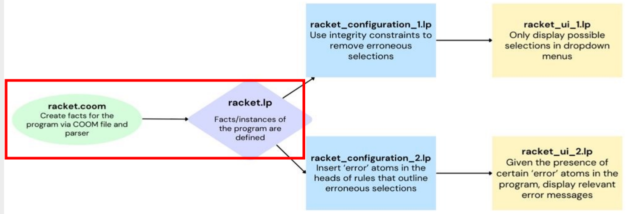
```
instances -s asp/solutions -t 100
instance-02-03.lp: failure in 0.205 seconds
instance-04-03.lp: failure in 0.017 seconds
instance-06-05.lp: failure in 0.016 seconds
instance-08-05.lp: failure in 0.017 seconds
instance-10-05.lp: failure in 0.016 seconds
instance-12-07.lp: failure in 0.017 seconds
instance-18-09.lp: failure in 0.016 seconds
instance-20-09.lp: failure in 0.017 seconds
instance-22-09.lp: failure in 0.016 seconds
instance-24-09.lp: failure in 0.016 seconds
instance-26-09.lp: failure in 0.017 seconds
instance-26-11.lp: failure in 0.017 seconds
instance-28-09.lp: failure in 0.024 seconds
instance-28-11.lp: failure in 0.018 seconds
instance-32-11.lp: failure in 0.018 seconds
instance-32-13.lp: failure in 0.018 seconds
instance-46-17.lp: failure in 0.018 seconds
instance-46-19.lp: failure in 0.018 seconds
instance-48-19.lp: failure in 0.018 seconds
instance-48-21.lp: failure in 0.018 seconds
FAILURE
```

Overview

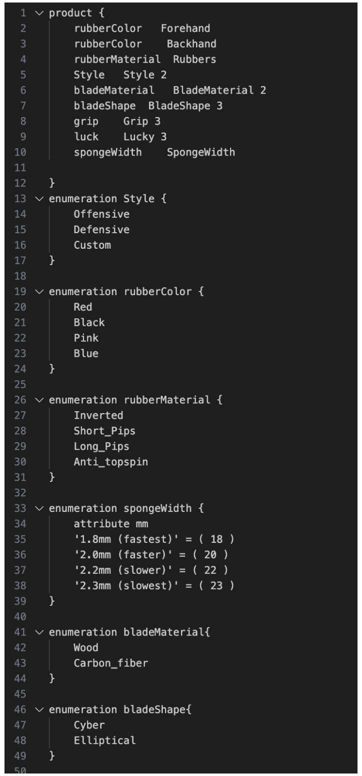


Project Pipeline
Method 1 (Default):
(Not allowing the selection of erroneous features in program)
racket.coom
Create facts for the program via COOM file and parser
racket.lp
Facts/instances of the program are defined
racket_configuration_1.lp
Use integrity constraints to remove erroneous selections
racket_ui_1.lp
Only display possible selections in dropdown menus
racket_configuration_2.lp
Insert 'error' atoms in the heads of rules that outline erroneous selections
racket_ui_2.lp
Given the presence of certain 'error' atoms in the program, display relevant error messages
Method 2 (Explanation):
(Generate informative error messages for erroneous selections)

Coom & lp


racket.coom
Create facts for the program via COOM file and parser
racket.lp
Facts/instances of the program are defined
racket_configuration_1.lp
Use integrity constraints to remove erroneous selections
racket_configuration_2.lp
Insert 'error' atoms in the heads of rules that outline erroneous selections
racket_ui_1.lp
Only display possible selections in dropdown menus
racket_ui_2.lp
Given the presence of certain 'error' atoms in the program, display relevant error messages

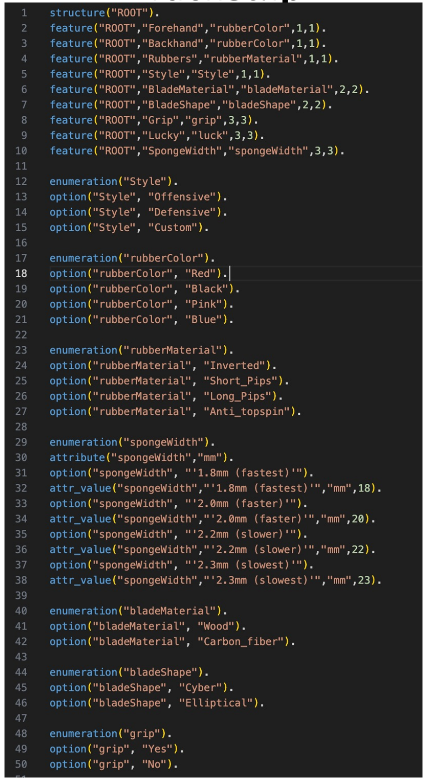
racket.coom



```
product {
    rubberColor   Forehand
    rubberColor    Backhand
    rubberMaterial  Rubbers
    Style   Style 2
    bladeMaterial   BladeMaterial 2
    bladeShape  BladeShape 3
    grip    Grip 3
    luck    Lucky 3
    spongeWidth    SpongeWidth

}
enumeration Style {
    Offensive
    Defensive
    Custom
}

enumeration rubberColor {
    Red
    Black
    Pink
    Blue
}

enumeration rubberMaterial {
    Inverted
    Short_Pips
    Long_Pips
    Anti_topspin
}

enumeration spongeWidth {
    attribute mm
    '1.8mm (fastest)' = ( 18 )
    '2.0mm (faster)' = ( 20 )
    '2.2mm (slower)' = ( 22 )
    '2.3mm (slowest)' = ( 23 )
}

enumeration bladeMaterial{
    Wood
    Carbon_fiber
}

enumeration bladeShape{
    Cyber
    Elliptical
}
```

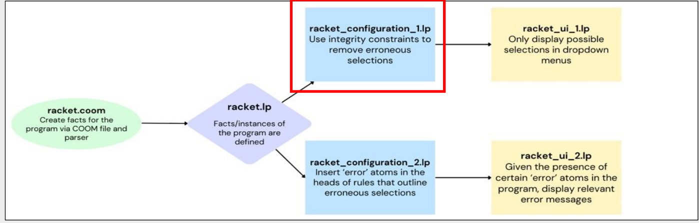
Use coom **parser** to translate racket.coom to racket.lp

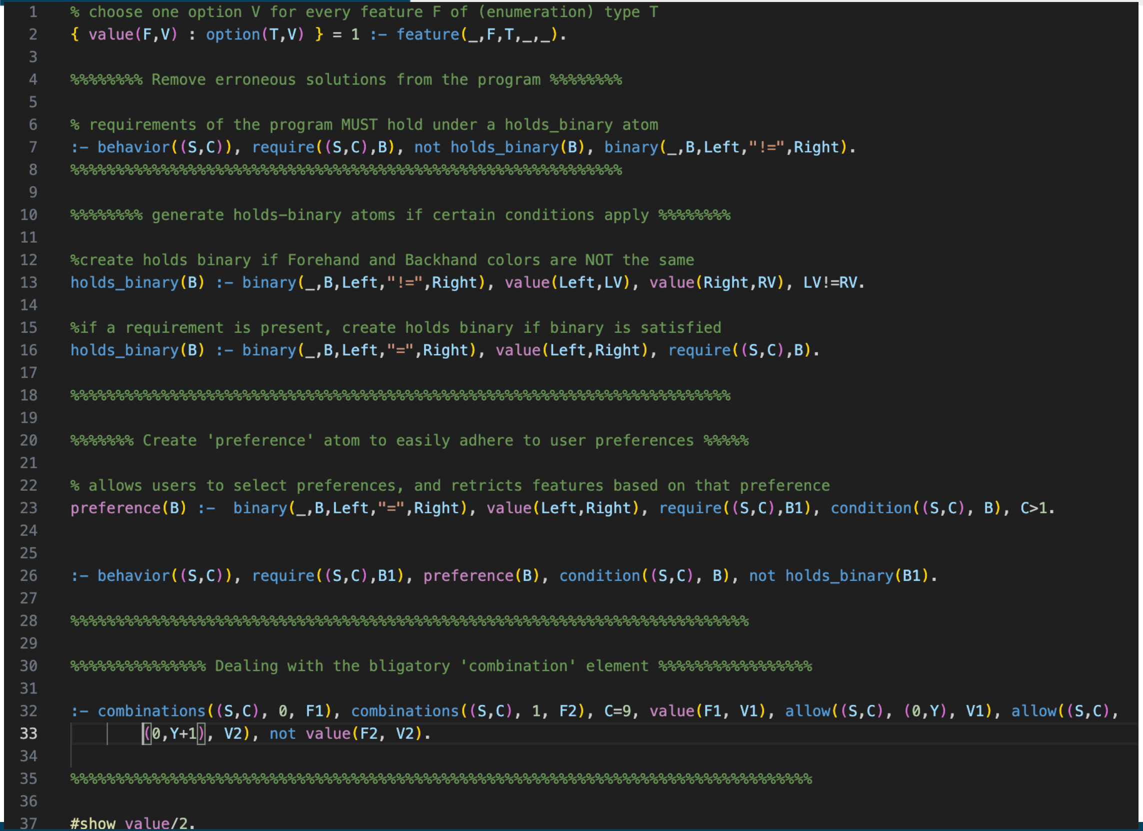
racket.lp



```
structure("ROOT").
feature("ROOT","Forehand","rubberColor",1,1).
feature("ROOT","Backhand","rubberColor",1,1).
feature("ROOT","Rubbers","rubberMaterial",1,1).
feature("ROOT","Style","Style",1,1).
feature("ROOT","BladeMaterial","bladeMaterial",2,2).
feature("ROOT","BladeShape","bladeShape",2,2).
feature("ROOT","Grip","grip",3,3).
feature("ROOT","Lucky","luck",3,3).
feature("ROOT","SpongeWidth","spongeWidth",3,3).

enumeration("Style").
option("Style", "Offensive").
option("Style", "Defensive").
option("Style", "Custom").

enumeration("rubberColor").
option("rubberColor", "Red").
option("rubberColor", "Black").
option("rubberColor", "Pink").
option("rubberColor", "Blue").

enumeration("rubberMaterial").
option("rubberMaterial", "Inverted").
option("rubberMaterial", "Short_Pips").
option("rubberMaterial", "Long_Pips").
option("rubberMaterial", "Anti_topspin").

enumeration("spongeWidth").
attribute("spongeWidth","mm").
option("spongeWidth", "'1.8mm (fastest)'").
attr_value("spongeWidth","'1.8mm (fastest)'","mm",18).
option("spongeWidth", "'2.0mm (faster)'").
attr_value("spongeWidth","'2.0mm (faster)'","mm",20).
option("spongeWidth", "'2.2mm (slower)'").
attr_value("spongeWidth","'2.2mm (slower)'","mm",22).
option("spongeWidth", "'2.3mm (slowest)'").
attr_value("spongeWidth","'2.3mm (slowest)'","mm",23).

enumeration("bladeMaterial").
option("bladeMaterial", "Wood").
option("bladeMaterial", "Carbon_fiber").

enumeration("bladeShape").
option("bladeShape", "Cyber").
option("bladeShape", "Elliptical").

enumeration("grip").
option("grip", "Yes").
option("grip", "No").
```

racket_configuration_1.lp


racket.coom
Create facts for the program via COOM file and parser
racket.lp
Facts/instances of the program are defined
racket_configuration_1.lp
Use integrity constraints to remove erroneous selections
racket_configuration_2.lp
Insert 'error' atoms in the heads of rules that outline erroneous selections
racket_ui_1.lp
Only display possible selections in dropdown menus
racket_ui_2.lp
Given the presence of certain 'error' atoms in the program, display relevant error messages



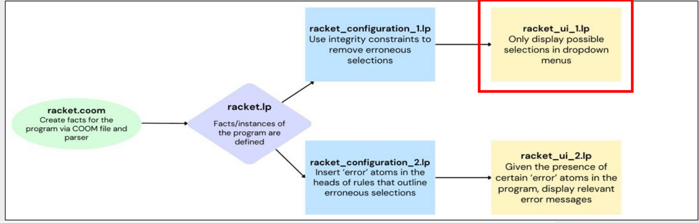
```
% choose one option V for every feature F of (enumeration) type T
{ value(F,V) : option(T,V) } = 1 :- feature(_,F,T,_,_).

%%%%%%%% Remove erroneous solutions from the program %%%%%%%%

% requirements of the program MUST hold under a holds_binary atom
:- behavior((S,C)), require((S,C),B), not holds_binary(B), binary(_,B,Left,"!=",Right).
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%% generate holds-binary atoms if certain conditions apply %%%%%%%%

%create holds binary if Forehand and Backhand colors are NOT the same
holds_binary(B) :- binary(_,B,Left,"!=",Right), value(Left,LV), value(Right,RV), LV!=RV.

%if a requirement is present, create holds binary if binary is satisfied
holds_binary(B) :- binary(_,B,Left,"=",Right), value(Left,Right), require((S,C),B).

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%% Create 'preference' atom to easily adhere to user preferences %%%%%

% allows users to select preferences, and retricts features based on that preference
preference(B) :-  binary(_,B,Left,"=",Right), value(Left,Right), require((S,C),B1), condition((S,C), B), C>1.


:- behavior((S,C)), require((S,C),B1), preference(B), condition((S,C), B), not holds_binary(B1).

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%% Dealing with the bligatory 'combination' element %%%%%%%%%%%%%%%%%

:- combinations((S,C), 0, F1), combinations((S,C), 1, F2), C=9, value(F1, V1), allow((S,C), (0,Y), V1), allow((S,C),
        (0,Y+1), V2), not value(F2, V2).

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

#show value/2.
```

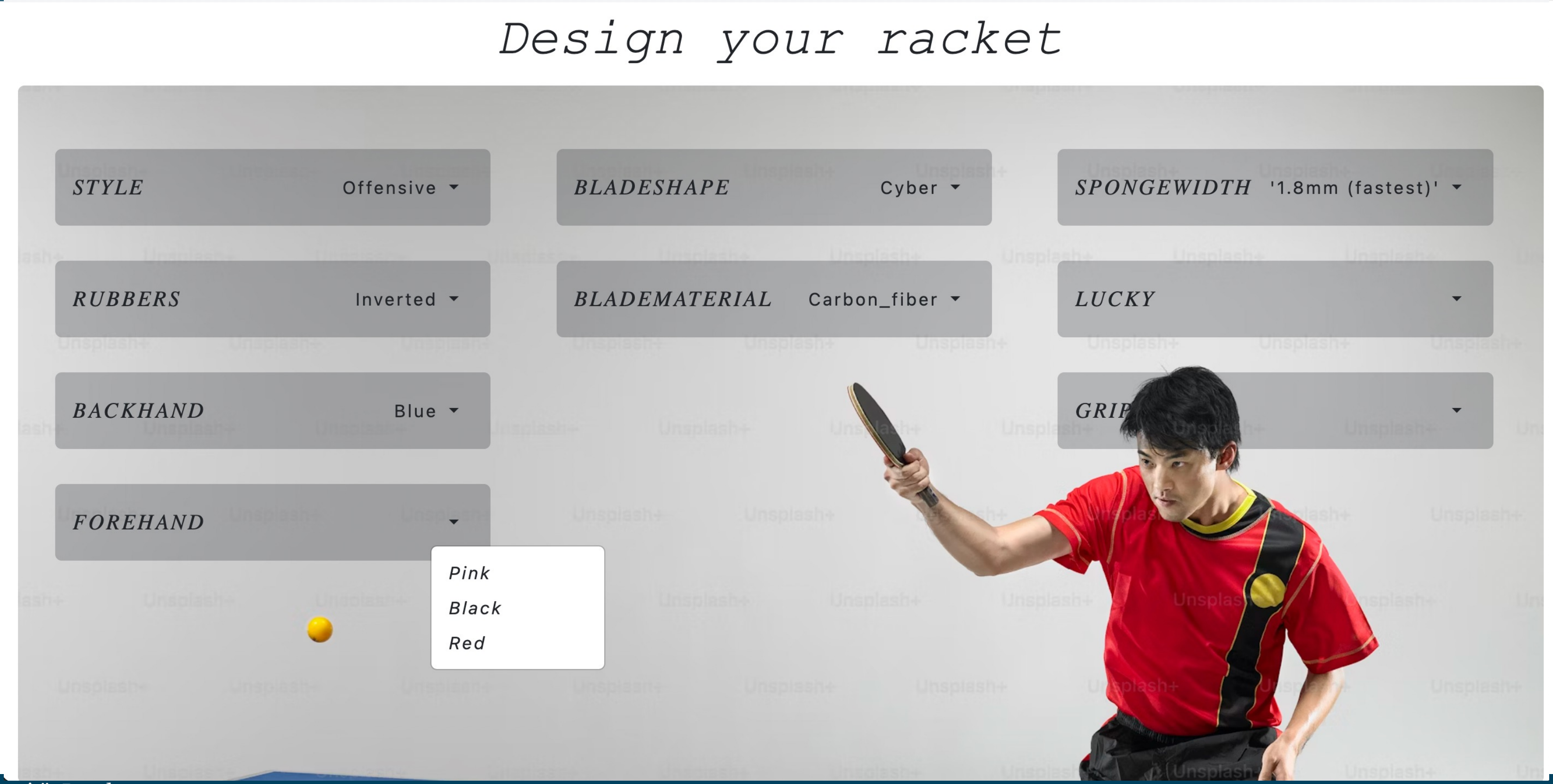
Generate possible solutions and Eliminate invalid ones

(e.g. remove solutions where *Forehand* and *Backhand* colors are identical)

Universität Potsdam

User Interface 1


racket.coom
Create facts for the program via COOM file and parser
racket.lp
Facts/instances of the program are defined
racket_configuration_1.lp
Use integrity constraints to remove erroneous selections
racket_ui_1.lp
Only display possible selections in dropdown menus
racket_configuration_2.lp
Insert 'error' atoms in the heads of rules that outline erroneous selections
racket_ui_2.lp
Given the presence of certain 'error' atoms in the program, display relevant error messages


Design your racket

STYLE Offensive

BLADESHAPE Cyber

SPONGEWIDTH '1.8mm (fastest)'

RUBBERS Inverted

BLADEMATERIAL Carbon_fiber

LUCKY

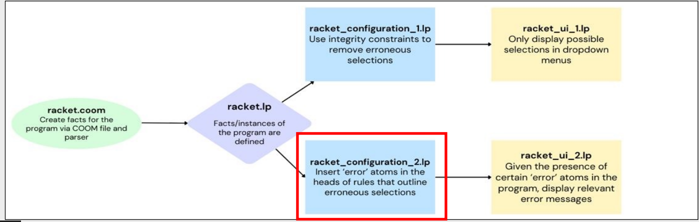
BACKHAND Blue

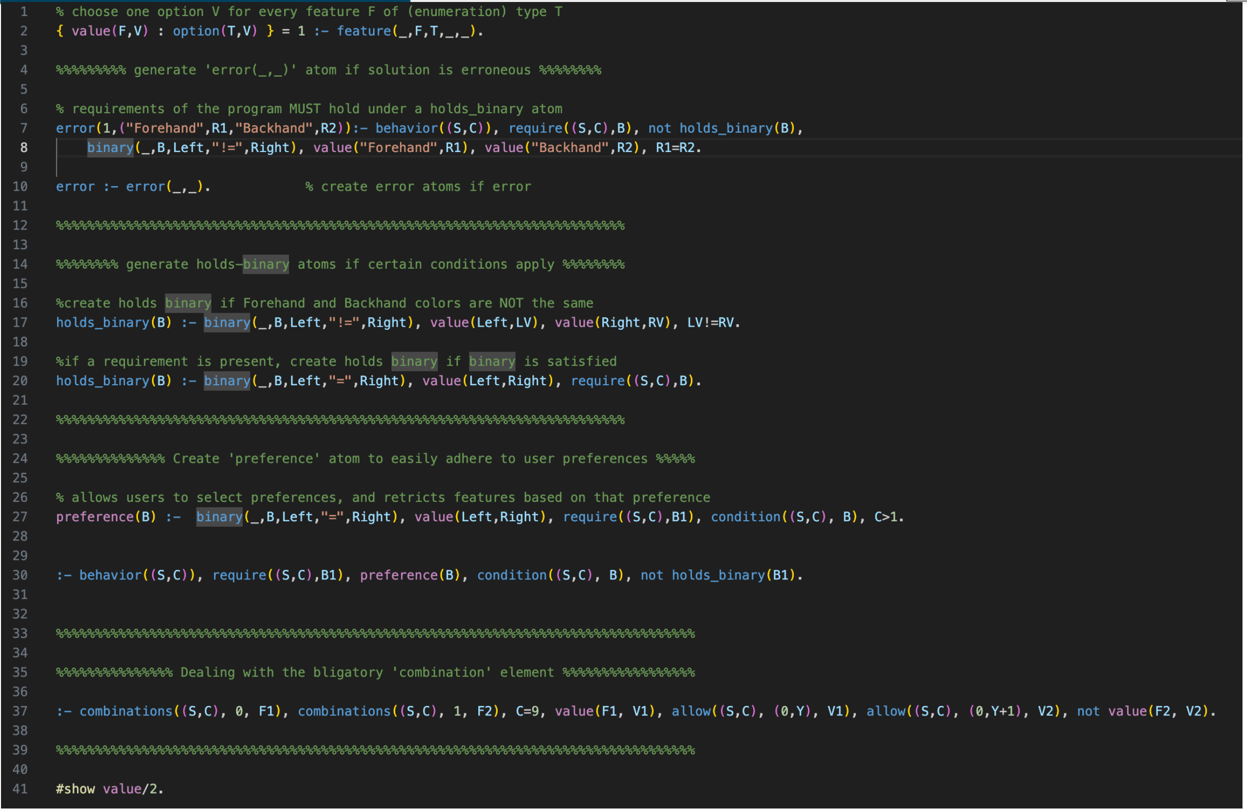
GRIP

FOREHAND

Universität Potsdam

racket_configuration_2.lp


racket.coom
Create facts for the program via COOM file and parser
racket.lp
Facts/instances of the program are defined
racket_configuration_1.lp
Use integrity constraints to remove erroneous selections
racket_ui_1.lp
Only display possible selections in dropdown menus
racket_configuration_2.lp
Insert 'error' atoms in the heads of rules that outline erroneous selections
racket_ui_2.lp
Given the presence of certain 'error' atoms in the program, display relevant error messages



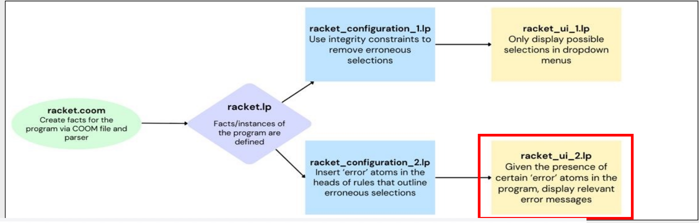
```
% choose one option V for every feature F of (enumeration) type T
{ value(F,V) : option(T,V) } = 1 :- feature(_,F,T,_,_).

%%%%%%%%% generate 'error(_,_)' atom if solution is erroneous %%%%%%%%

% requirements of the program MUST hold under a holds_binary atom
error(1,("Forehand",R1,"Backhand",R2)):- behavior((S,C)), require((S,C),B), not holds_binary(B),
    binary(_,B,Left,"!=",Right), value("Forehand",R1), value("Backhand",R2), R1=R2.

error :- error(_,_).           % create error atoms if error

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%% generate holds-binary atoms if certain conditions apply %%%%%%%%

%create holds binary if Forehand and Backhand colors are NOT the same
holds_binary(B) :- binary(_,B,Left,"!=",Right), value(Left,LV), value(Right,RV), LV!=RV.

%if a requirement is present, create holds binary if binary is satisfied
holds_binary(B) :- binary(_,B,Left,"=",Right), value(Left,Right), require((S,C),B).

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%% Create 'preference' atom to easily adhere to user preferences %%%%%

% allows users to select preferences, and retricts features based on that preference
preference(B) :-  binary(_,B,Left,"=",Right), value(Left,Right), require((S,C),B1), condition((S,C), B), C>1.


:- behavior((S,C)), require((S,C),B1), preference(B), condition((S,C), B), not holds_binary(B1).


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%% Dealing with the bligatory 'combination' element %%%%%%%%%%%%%%%%%

:- combinations((S,C), 0, F1), combinations((S,C), 1, F2), C=9, value(F1, V1), allow((S,C), (0,Y), V1), allow((S,C), (0,Y+1), V2), not value(F2, V2).

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

#show value/2.
```

Generate possible solutions *and* generate error atoms for invalid ones

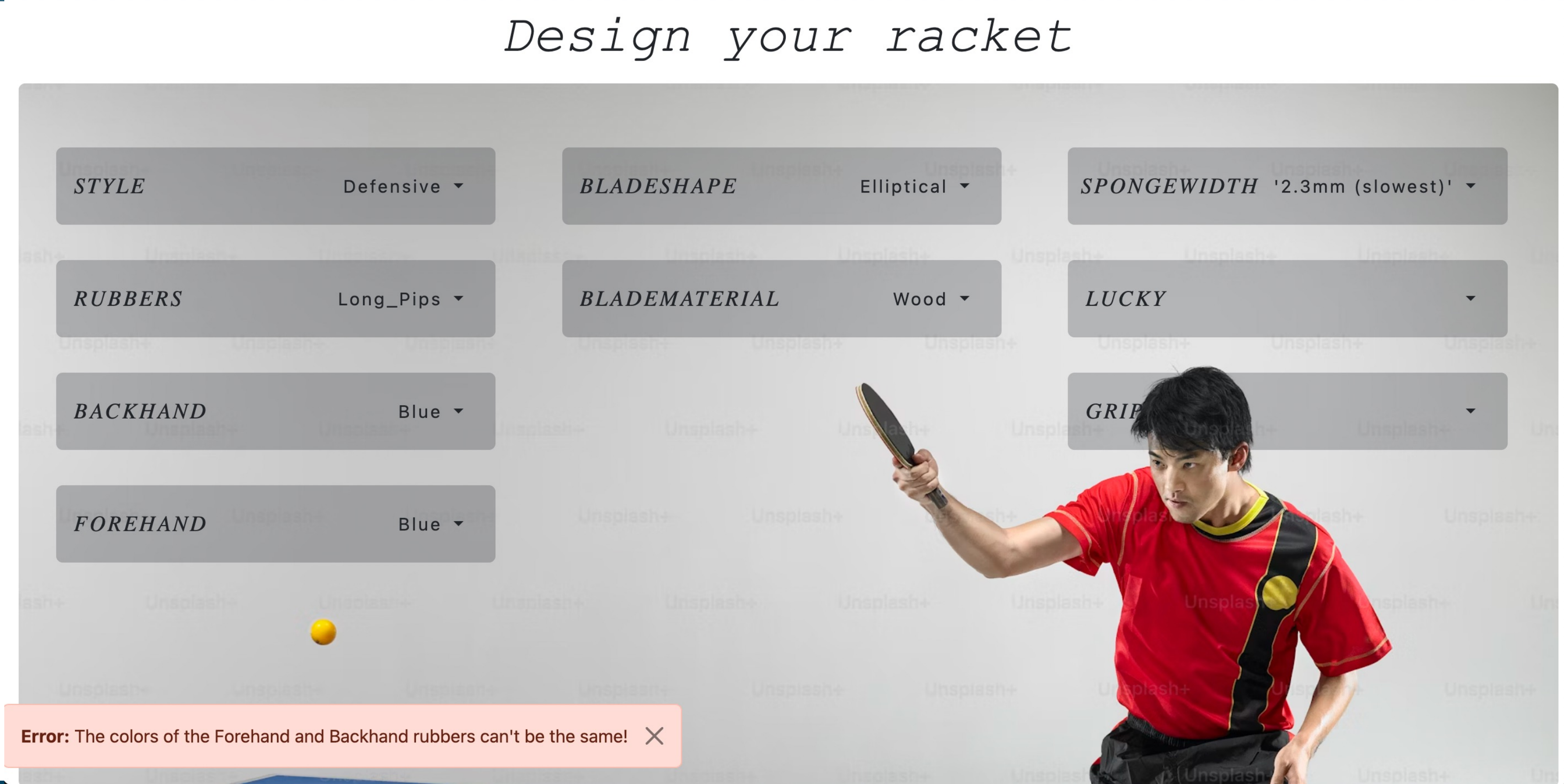
(e.g. if the same color is selected for both *Fronthand* and *Backhand* colors, an *error(_,_)* is created)

Universität Potsdam

User Interface 2



racket.coom
Create facts for the program via COOM file and parser
racket.lp
Facts/instances of the program are defined
racket_configuration_1.lp
Use integrity constraints to remove erroneous selections
racket_ui_1.lp
Only display possible selections in dropdown menus
racket_configuration_2.lp
Insert 'error' atoms in the heads of rules that outline erroneous selections
racket_ui_2.lp
Given the presence of certain 'error' atoms in the program, display relevant error messages



Design your racket

STYLE Defensive

BLADESHAPE Elliptical

SPONGEWIDTH '2.3mm (slowest)'

RUBBERS Long_Pips

BLADEMATERIAL Wood

LUCKY

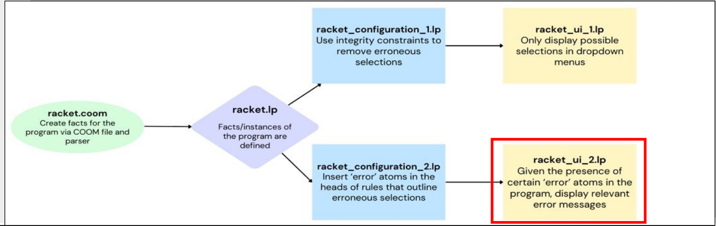
BACKHAND Blue

GRIP

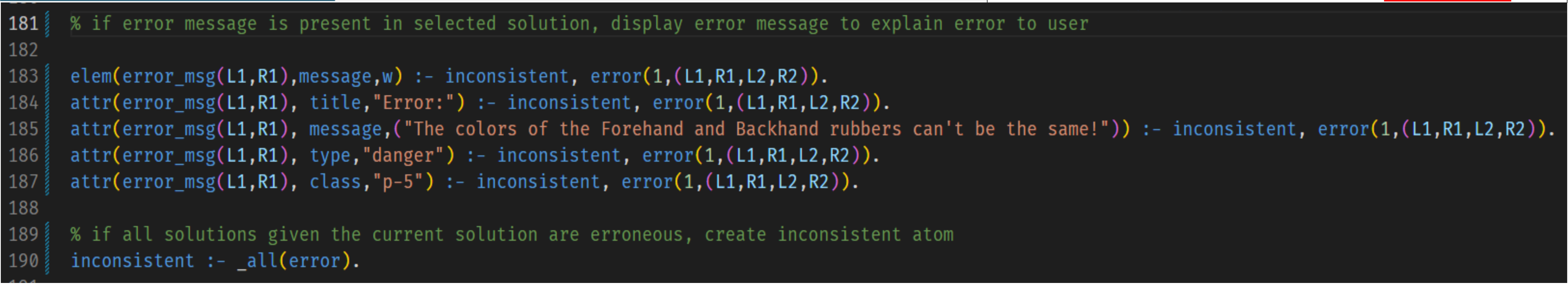
FOREHAND Blue

Error: The colors of the Forehand and Backhand rubbers can't be the same!

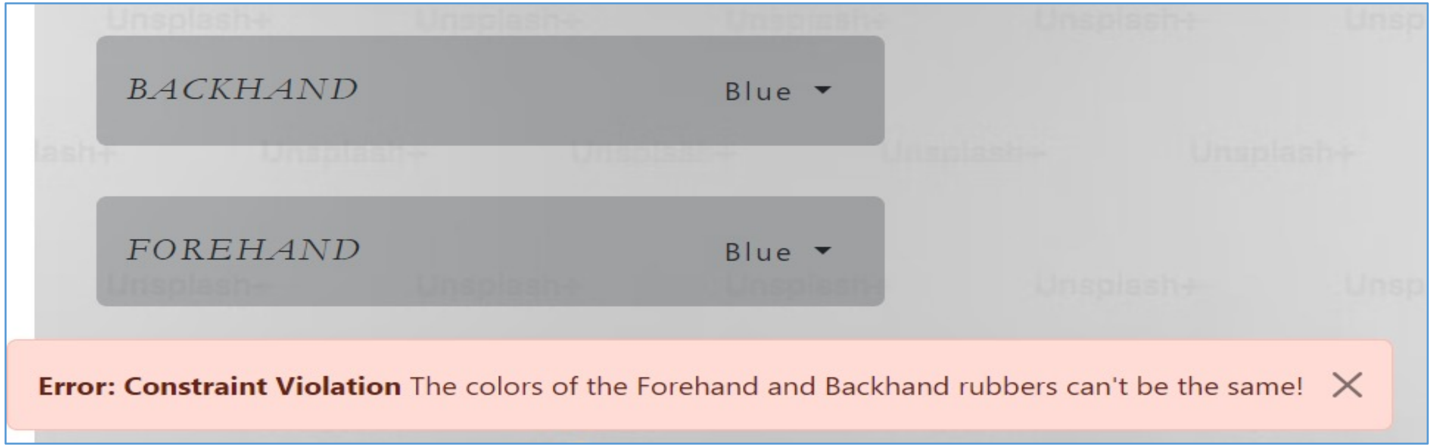
User Interface 2



racket.coom
Create facts for the program via COOM file and parser
racket.lp
Facts/instances of the program are defined
racket_configuration_1.lp
Use integrity constraints to remove erroneous selections
racket_ui_1.lp
Only display possible selections in dropdown menus
racket_configuration_2.lp
Insert 'error' atoms in the heads of rules that outline erroneous selections
racket_ui_2.lp
Given the presence of certain 'error' atoms in the program, display relevant error messages



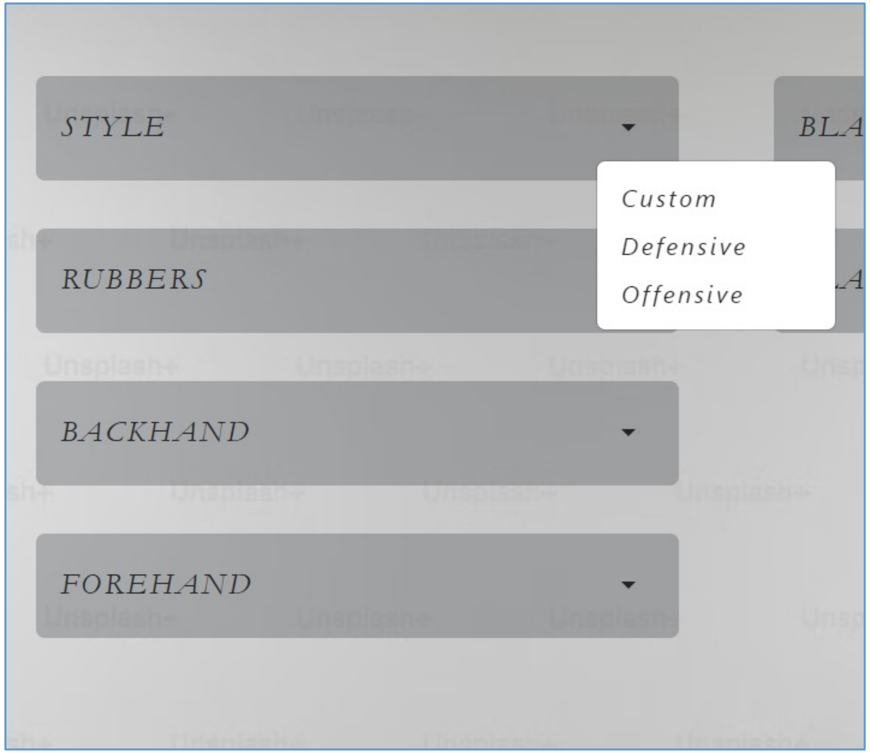
```
% if error message is present in selected solution, display error message to explain error to user

elem(error_msg(L1,R1),message,w) :- inconsistent, error(1,(L1,R1,L2,R2)).
attr(error_msg(L1,R1), title,"Error:") :- inconsistent, error(1,(L1,R1,L2,R2)).
attr(error_msg(L1,R1), message,("The colors of the Forehand and Backhand rubbers can't be the same!")) :- inconsistent, error(1,(L1,R1,L2,R2)).
attr(error_msg(L1,R1), type,"danger") :- inconsistent, error(1,(L1,R1,L2,R2)).
attr(error_msg(L1,R1), class,"p-5") :- inconsistent, error(1,(L1,R1,L2,R2)).

% if all solutions given the current solution are erroneous, create inconsistent atom
inconsistent :- _all(error).
```



Error: Constraint Violation The colors of the Forehand and Backhand rubbers can't be the same!

Universität Potsdam

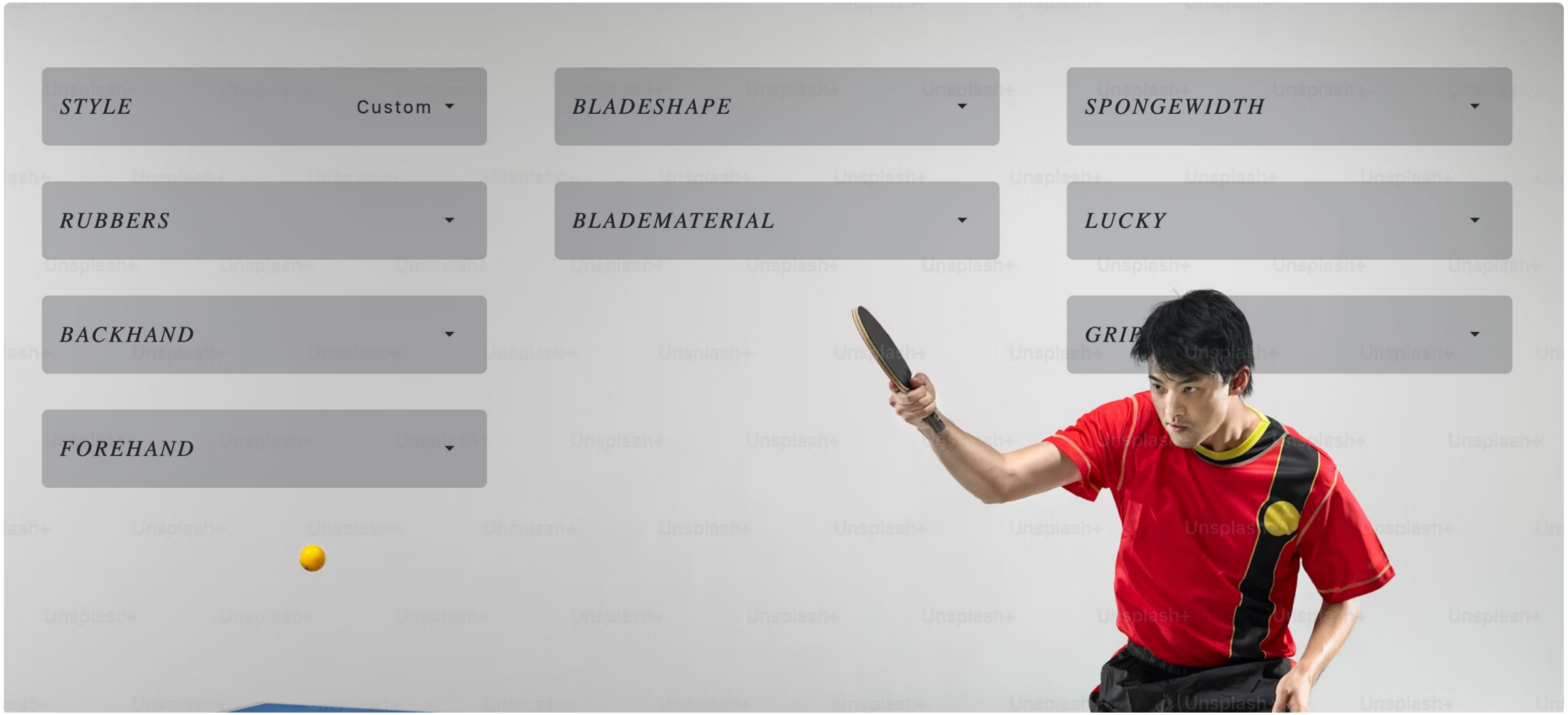
Preferences



- Preferences (Playing style):
  1. Custom
  2. Defensive
  3. Offensive

Preferences

Preference (Playing style): Custom



STYLE Custom

BLADESHAPE

SPONGEWIDTH

RUBBERS

BLADEMATERIAL

LUCKY

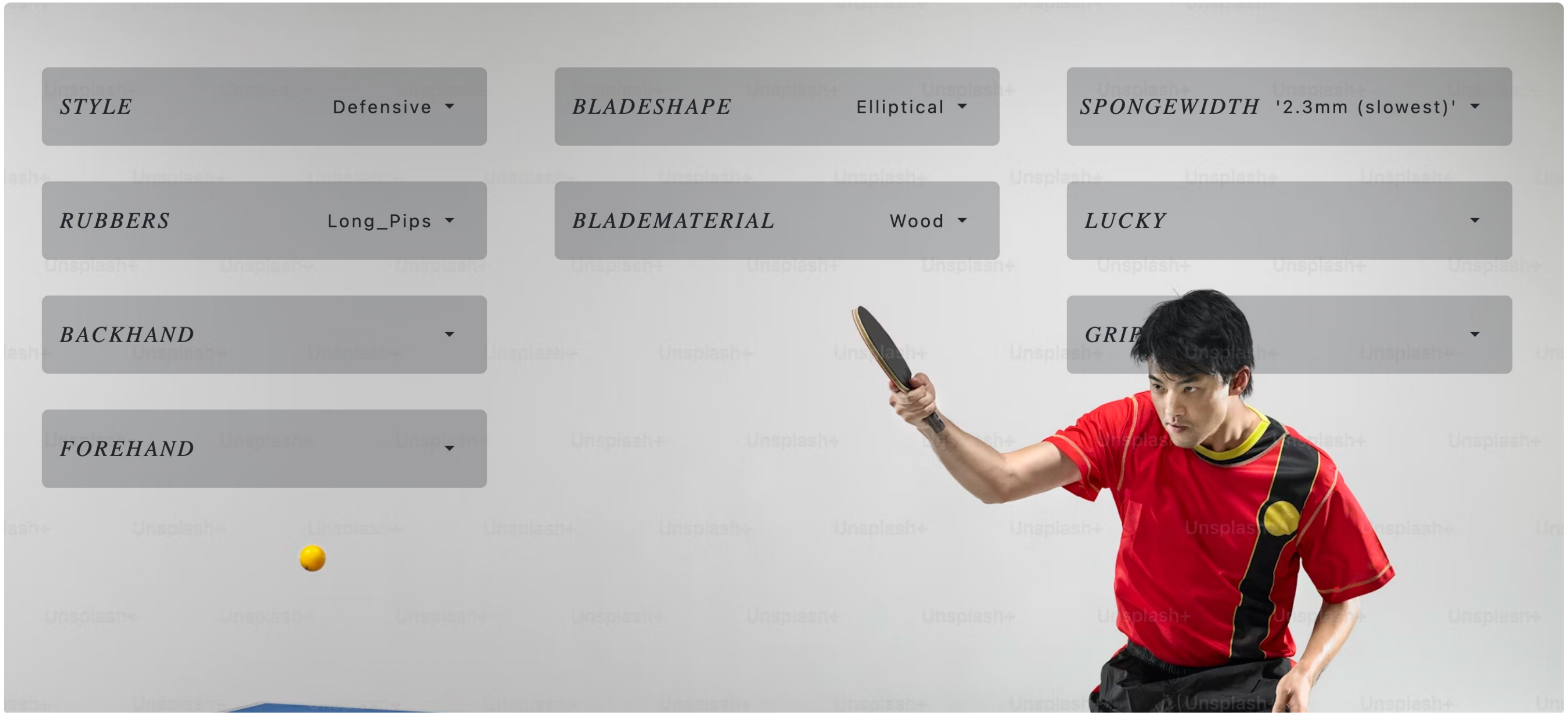
BACKHAND

GRIP

FOREHAND

Preferences

Preference (Playing style): Defensive



STYLE Defensive

BLADESHAPE Elliptical

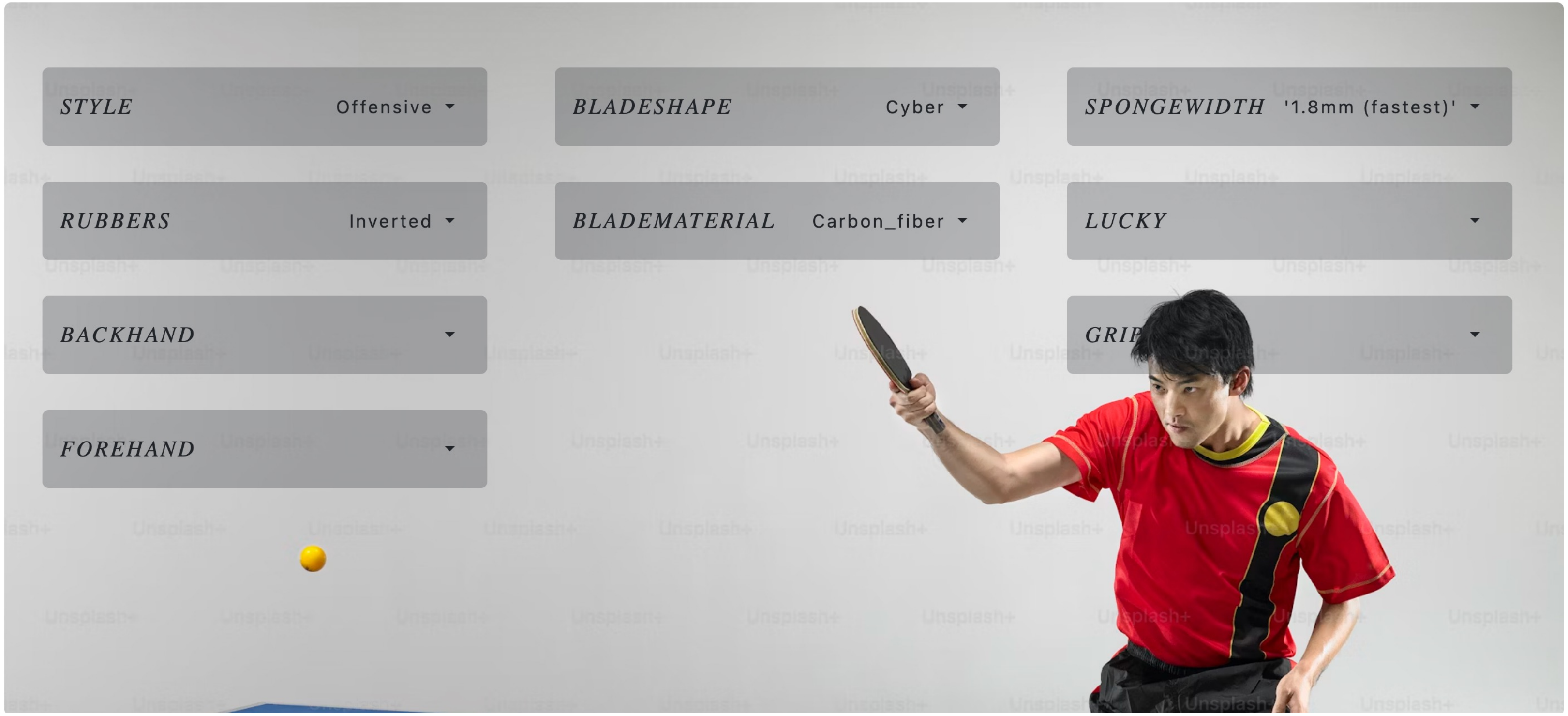
SPONGEWIDTH '2.3mm (slowest)'

RUBBERS Long_Pips

BLADEMATERIAL Wood

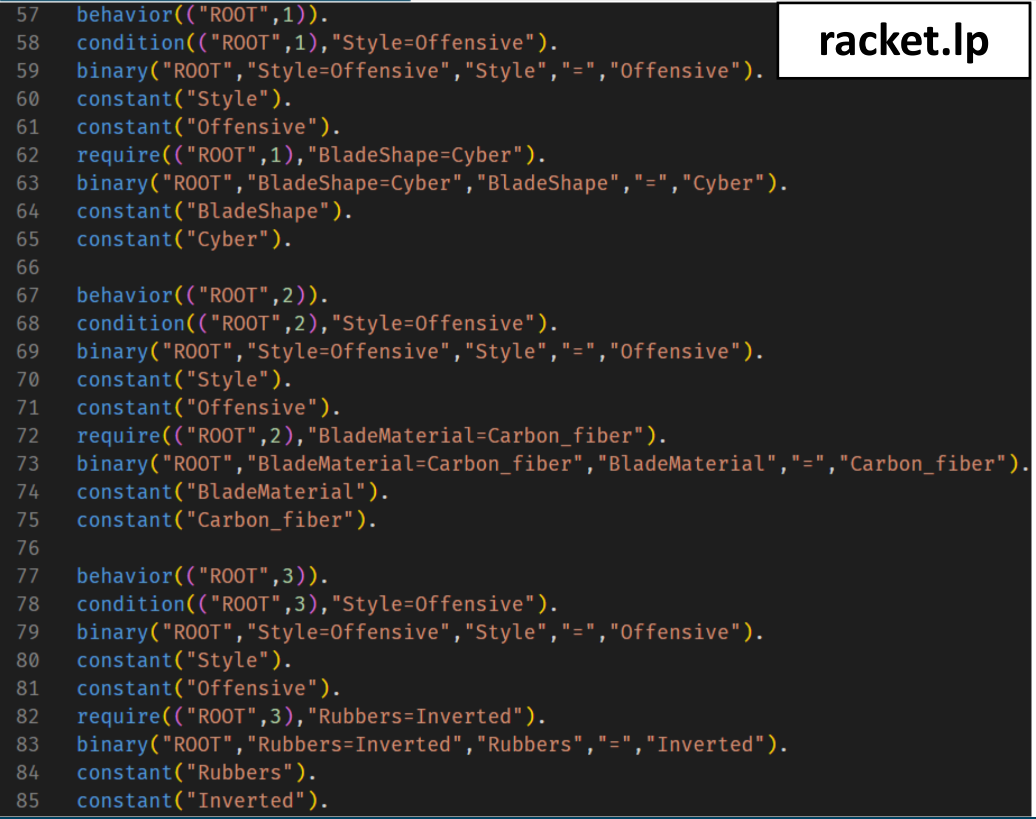
Preferences

Preference (Playing style): Offensive



STYLE	Offensive
BLADESHAPE	Cyber
SPONGEWIDTH	'1.8mm (fastest)'
RUBBERS	Inverted
BLADEMATERIAL	Carbon_fiber
LUCKY	
BACKHAND	
GRIP	
FOREHAND	

Preferences: The code



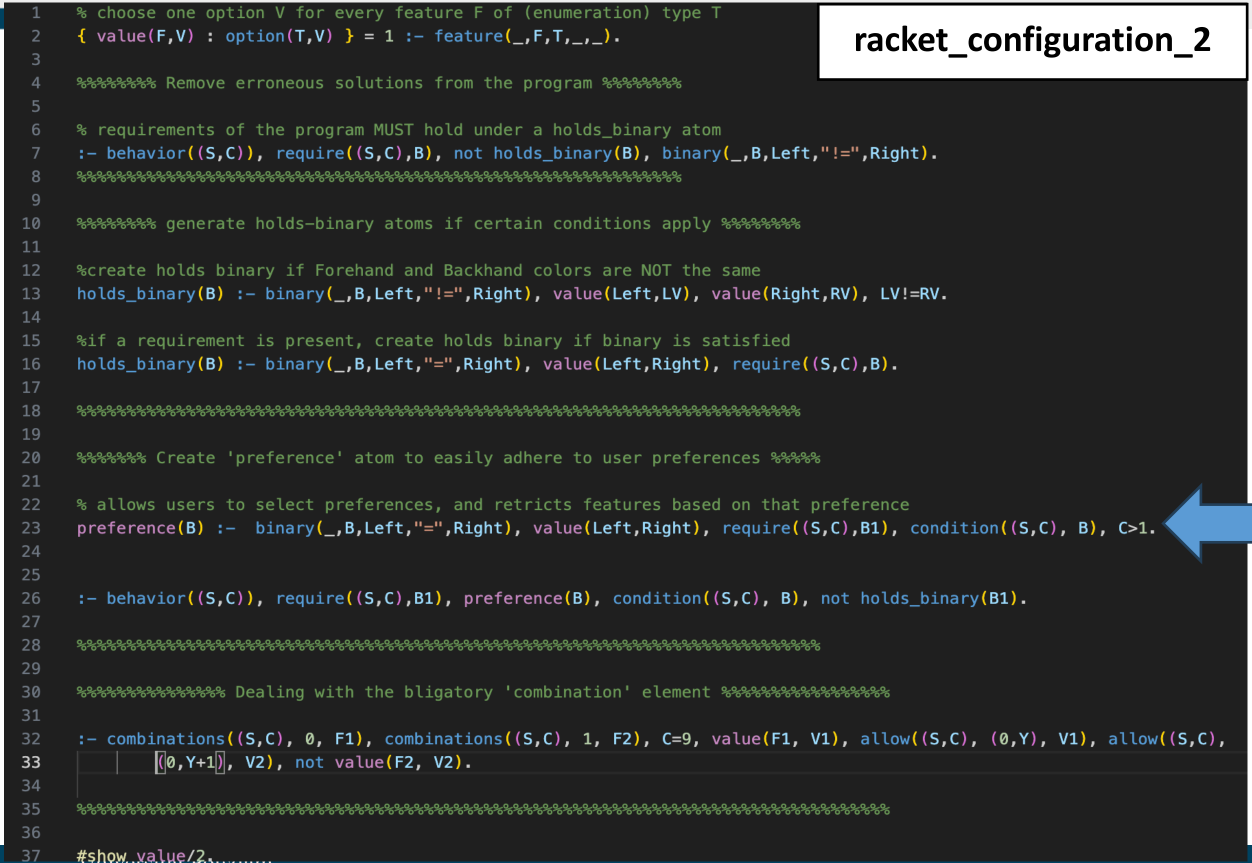
racket.lp

```
behavior(("ROOT",1)).
condition(("ROOT",1),"Style=Offensive").
binary("ROOT","Style=Offensive","Style","=","Offensive").
constant("Style").
constant("Offensive").
require(("ROOT",1),"BladeShape=Cyber").
binary("ROOT","BladeShape=Cyber","BladeShape","=","Cyber").
constant("BladeShape").
constant("Cyber").

behavior(("ROOT",2)).
condition(("ROOT",2),"Style=Offensive").
binary("ROOT","Style=Offensive","Style","=","Offensive").
constant("Style").
constant("Offensive").
require(("ROOT",2),"BladeMaterial=Carbon_fiber").
binary("ROOT","BladeMaterial=Carbon_fiber","BladeMaterial","=","Carbon_fiber").
constant("BladeMaterial").
constant("Carbon_fiber").

behavior(("ROOT",3)).
condition(("ROOT",3),"Style=Offensive").
binary("ROOT","Style=Offensive","Style","=","Offensive").
constant("Style").
constant("Offensive").
require(("ROOT",3),"Rubbers=Inverted").
binary("ROOT","Rubbers=Inverted","Rubbers","=","Inverted").
constant("Rubbers").
constant("Inverted").
```

Step 1

Define relationships between preferences and features in racket.lp

Preferences: The code



racket_configuration_2

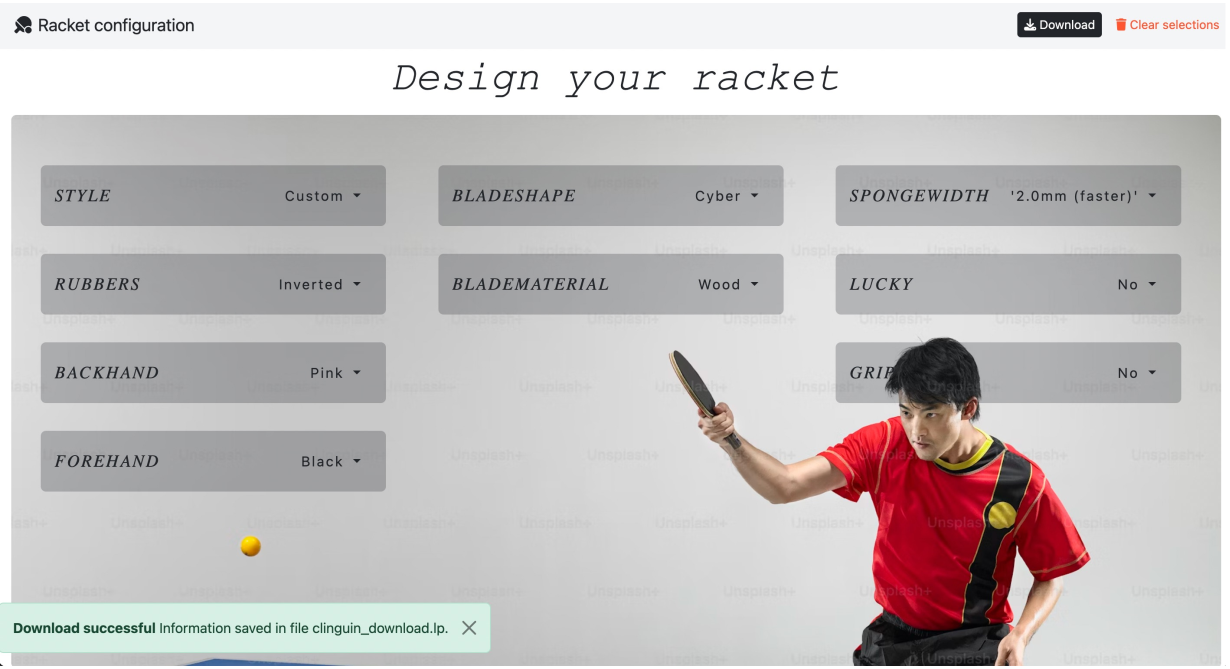
```
% choose one option V for every feature F of (enumeration) type T
{ value(F,V) : option(T,V) } = 1 :- feature(_,F,T,_,_).

%%%%%%%% Remove erroneous solutions from the program %%%%%%%%

% requirements of the program MUST hold under a holds_binary atom
:- behavior((S,C)), require((S,C),B), not holds_binary(B), binary(_,B,Left,"!=",Right).
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%% generate holds-binary atoms if certain conditions apply %%%%%%%%

%create holds binary if Forehand and Backhand colors are NOT the same
holds_binary(B) :- binary(_,B,Left,"!=",Right), value(Left,LV), value(Right,RV), LV!=RV.

%if a requirement is present, create holds binary if binary is satisfied
holds_binary(B) :- binary(_,B,Left,"=",Right), value(Left,Right), require((S,C),B).

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%% Create 'preference' atom to easily adhere to user preferences %%%%%

% allows users to select preferences, and retricts features based on that preference
preference(B) :-  binary(_,B,Left,"=",Right), value(Left,Right), require((S,C),B1), condition((S,C), B), C>1.


:- behavior((S,C)), require((S,C),B1), preference(B), condition((S,C), B), not holds_binary(B1).

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%% Dealing with the bligatory 'combination' element %%%%%%%%%%%%%%%%%

:- combinations((S,C), 0, F1), combinations((S,C), 1, F2), C=9, value(F1, V1), allow((S,C), (0,Y), V1), allow((S,C),
        (0,Y+1), V2), not value(F2, V2).

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

#show value/2.
```

Step 2

Given the preference-feature relationships, configure information into relevant atoms

If preference(A) == *true*
then
set(A) of features == *true*

Download

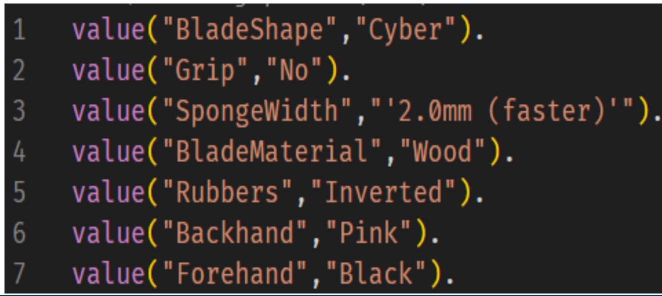

Racket configuration

Design your racket

STYLE	Custom
BLADESHAPE	Cyber
SPONGEWIDTH	'2.0mm (faster)'
RUBBERS	Inverted
BLADEMATERIAL	Wood
LUCKY	No
BACKHAND	Pink
GRIP	No
FOREHAND	Black

Download successful Information saved in file clinguin_download.lp.

Downloading a solution.


```
value("BladeShape","Cyber").
value("Grip","No").
value("SpongeWidth","'2.0mm (faster)'").
value("BladeMaterial","Wood").
value("Rubbers","Inverted").
value("Backhand","Pink").
value("Forehand","Black").
```