

# Trello\_Project\_Management\_System\_

## 1.0

Generated by Doxygen 1.13.2

<b>1 Data Structure Index</b>	<b>1</b>
1.1 Data Structures	1
<b>2 File Index</b>	<b>1</b>
2.1 File List	1
<b>3 Data Structure Documentation</b>	<b>2</b>
3.1 CARD Struct Reference	2
3.1.1 Detailed Description	2
3.1.2 Field Documentation	2
3.2 LIST Struct Reference	3
3.2.1 Detailed Description	3
3.2.2 Field Documentation	3
3.3 USER Struct Reference	4
3.3.1 Detailed Description	4
3.3.2 Field Documentation	4
<b>4 File Documentation</b>	<b>4</b>
4.1 cards.c File Reference	4
4.1.1 Function Documentation	5
4.2 cards.h File Reference	7
4.2.1 Detailed Description	8
4.2.2 Macro Definition Documentation	8
4.2.3 Typedef Documentation	8
4.2.4 Function Documentation	8
4.3 cards.h	10
4.4 lists.c File Reference	11
4.4.1 Function Documentation	11
4.5 lists.h File Reference	13
4.5.1 Detailed Description	14
4.5.2 Macro Definition Documentation	14
4.5.3 Typedef Documentation	15
4.5.4 Function Documentation	15
4.6 lists.h	16
4.7 search.c File Reference	16
4.7.1 Function Documentation	17
4.8 search.h File Reference	17
4.8.1 Detailed Description	18
4.8.2 Function Documentation	18
4.9 search.h	18
4.10 user_entry.c File Reference	19
4.10.1 Function Documentation	19
4.11 user_entry.h File Reference	19

4.11.1 Detailed Description . . . . .	20
4.11.2 Macro Definition Documentation . . . . .	20
4.11.3 Typedef Documentation . . . . .	20
4.11.4 Function Documentation . . . . .	20
4.12 user_entry.h . . . . .	20
<b>Index</b>	<b>21</b>

## 1 Data Structure Index

### 1.1 Data Structures

Here are the data structures with brief descriptions:

<b>CARD</b>	
Represents a Card in Trello_Project_Management_System	2
<b>LIST</b>	
Represents a List in Trello_Project_Management_System	3
<b>USER</b>	
Represents a identity of a user	4

## 2 File Index

### 2.1 File List

Here is a list of all files with brief descriptions:

<b>cards.c</b>	4
<b>cards.h</b>	
Header for managing the cards in the Tello_Project_Management_System	7
<b>lists.c</b>	11
<b>lists.h</b>	
Header for managing the lists in the Tello_Project_Management_System	13
<b>search.c</b>	16
<b>search.h</b>	
Header for searching the cards or lists in the Tello_Project_Management_System	17
<b>user_entry.c</b>	19
<b>user_entry.h</b>	
Header for managing user entry	19

## 3 Data Structure Documentation

### 3.1 CARD Struct Reference

Represents a Card in Trello\_Project\_Management\_System.

```
#include <cards.h>
```

#### Data Fields

- char [name](#) [[MAX\\_NAME\\_LENGTH](#)]
- char [description](#) [[MAX\\_DESCRIPTION\\_LENGTH](#)]
- struct [CARD](#) \* [next\\_card](#)
- struct [CARD](#) \* [prev\\_card](#)

#### 3.1.1 Detailed Description

Represents a Card in Trello\_Project\_Management\_System.

#### 3.1.2 Field Documentation

##### description

```
char description[MAX\_DESCRIPTION\_LENGTH]
```

[CARD](#) description

##### name

```
char name[MAX\_NAME\_LENGTH]
```

[CARD](#) name

##### next\_card

```
struct CARD* next_card
```

A pointer to the next card

##### prev\_card

```
struct CARD* prev_card
```

A pointer to the previous card

The documentation for this struct was generated from the following file:

- [cards.h](#)

## 3.2 LIST Struct Reference

Represents a List in Trello\_Project\_Management\_System.

```
#include <lists.h>
```

### Data Fields

- char [name](#) [[MAX\\_NAME\\_LENGTH](#)]
- [CARD](#) \* [CARD](#)
- struct [LIST](#) \* [next\\_List](#)
- struct [LIST](#) \* [prev\\_List](#)

### 3.2.1 Detailed Description

Represents a List in Trello\_Project\_Management\_System.

### 3.2.2 Field Documentation

#### CARD

[CARD](#)\* [CARD](#)

A pointer to the first card of the list

#### name

char [name](#) [[MAX\\_NAME\\_LENGTH](#)]

[LIST](#) [name](#)

#### next\_List

struct [LIST](#)\* [next\\_List](#)

A pointer to the next list

#### prev\_List

struct [LIST](#)\* [prev\\_List](#)

A pointer to the previous list

The documentation for this struct was generated from the following file:

- [lists.h](#)

### 3.3 USER Struct Reference

Represents a identity of a user.

```
#include <user_entry.h>
```

#### Data Fields

- char [name](#) [[MAX\\_LENGTH](#)]
- char [email\\_id](#) [[MAX\\_LENGTH](#)]

#### 3.3.1 Detailed Description

Represents a identity of a user.

#### 3.3.2 Field Documentation

##### **email\_id**

```
char email_id[MAX\_LENGTH]
```

[USER](#) email id

##### **name**

```
char name[MAX\_LENGTH]
```

[USER](#) name

The documentation for this struct was generated from the following file:

- [user\\_entry.h](#)

## 4 File Documentation

### 4.1 cards.c File Reference

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <stdbool.h>
#include "cards.h"
```

## Functions

- `CARD * create_Card` (char name[`MAX_NAME_LENGTH`], char description[`MAX_DESCRIPTION_LENGTH`])  
*Creates a new card in the system.*
- void `add_a_Card` (`CARD *head`)  
*Insert a new card after the existing one.*
- `CARD * delete_a_Card` (`CARD *head`)  
*Delete any of the existing card from the system.*
- `CARD * delete_First_Card` (`CARD *head`)  
*Delete the very first card from the system.*
- `CARD * delete_Intermediate_or_Last_Card` (`CARD *head`)  
*Delete any of the existing card from the system without the first one.*
- bool `checkList` ()  
*Check whether the task of this card is done or not.*

### 4.1.1 Function Documentation

#### `add_a_Card()`

```
void add_a_Card (  
    CARD * head)
```

Insert a new card after the existing one.

##### Parameters

<i>head</i>	The pointer to the very first card
-------------	------------------------------------

#### `checkList()`

```
bool checkList ()
```

Check whether the task of this card is done or not.

##### Returns

DONE if the task has been completed, NOT\_DONE if that is incomplete

#### `create_Card()`

```
CARD * create_Card (  
    char name[MAX_NAME_LENGTH],  
    char description[MAX_DESCRIPTION_LENGTH])
```

Creates a new card in the system.

**Parameters**

<i>name</i>	The unique name of the card
<i>description</i>	The about information of the new card

**Returns**

A pointer to the new card

**delete\_a\_Card()**

```
CARD * delete_a_Card (  
    CARD * head)
```

Delete any of the existing card from the system.

**Parameters**

<i>head</i>	The pointer to the very first card
-------------	------------------------------------

**Returns**

A pointer to the new first card, if the very first card removed

**delete\_First\_Card()**

```
CARD * delete_First_Card (  
    CARD * head)
```

Delete the very first card from the system.

**Parameters**

<i>head</i>	The pointer to the existing first card
-------------	--

**Returns**

A pointer to the new first card

**delete\_Intermediate\_or\_Last\_Card()**

```
CARD * delete_Intermediate_or_Last_Card (  
    CARD * head)
```

Delete any of the existing card from the system without the first one.



## Parameters

<i>head</i>	The pointer to the very first card
-------------	------------------------------------

## Returns

A pointer to the first card

## 4.2 cards.h File Reference

Header for managing the cards in the Tello\_Project\_Management\_System.

```
#include <stdbool.h>
```

## Data Structures

- struct [CARD](#)  
*Represents a Card in Trello\_Project\_Management\_System.*

## Macros

- #define [MAX\\_DESCRIPTION\\_LENGTH](#) 250  
*Maximum number of characters allowed for writing description.*
- #define [MAX\\_NAME\\_LENGTH](#) 30  
*Maximum number of characters allowed for writing name.*
- #define [DONE](#) 1  
*To make the code readable, function will return DONE means 1.*
- #define [NOT\\_DONE](#) 0  
*To make the code readable, function will return NOT\_DONE means 0.*

## Typedefs

- typedef struct CARD [CARD](#)  
*Represents a Card in Trello\_Project\_Management\_System.*

## Functions

- [CARD \\* create\\_Card](#) (char name[[MAX\\_NAME\\_LENGTH](#)], char description[[MAX\\_DESCRIPTION\\_LENGTH](#)])  
*Creates a new card in the system.*
- void [add\\_a\\_Card](#) ([CARD \\*head](#))  
*Insert a new card after the existing one.*
- [CARD \\* delete\\_a\\_Card](#) ([CARD \\*head](#))  
*Delete any of the existing card from the system.*
- [CARD \\* delete\\_First\\_Card](#) ([CARD \\*head](#))  
*Delete the very first card from the system.*
- [CARD \\* delete\\_Intermediate\\_or\\_Last\\_Card](#) ([CARD \\*head](#))  
*Delete any of the existing card from the system without the first one.*
- bool [checkList](#) ()  
*Check whether the task of this card is done or not.*

#### 4.2.1 Detailed Description

Header for managing the cards in the Tello\_Project\_Management\_System.

Author

Data\_Structures\_Project\_Group\_01

#### 4.2.2 Macro Definition Documentation

##### DONE

```
#define DONE 1
```

To make the code readable, function will return DONE means 1.

##### MAX\_DESCRIPTION\_LENGTH

```
#define MAX_DESCRIPTION_LENGTH 250
```

Maximum number of characters allowed for writing description.

##### MAX\_NAME\_LENGTH

```
#define MAX_NAME_LENGTH 30
```

Maximum number of characters allowed for writing name.

##### NOT\_DONE

```
#define NOT_DONE 0
```

To make the code readable, function will return NOT\_DONE means 0.

#### 4.2.3 Typedef Documentation

##### CARD

```
typedef struct CARD CARD
```

Represents a Card in Trello\_Project\_Management\_System.

#### 4.2.4 Function Documentation

##### add\_a\_Card()

```
void add_a_Card (  
    CARD * head)
```

Insert a new card after the existing one.

## Parameters

<i>head</i>	The pointer to the very first card
-------------	------------------------------------

**checkList()**

```
bool checkList ()
```

Check whether the task of this card is done or not.

## Returns

DONE if the task has been completed, NOT\_DONE if that is incomplete

**create\_Card()**

```
CARD * create_Card (  
    char name[MAX_NAME_LENGTH],  
    char description[MAX_DESCRIPTION_LENGTH])
```

Creates a new card in the system.

## Parameters

<i>name</i>	The unique name of the card
<i>description</i>	The about information of the new card

## Returns

A pointer to the new card

**delete\_a\_Card()**

```
CARD * delete_a_Card (  
    CARD * head)
```

Delete any of the existing card from the system.

## Parameters

<i>head</i>	The pointer to the very first card
-------------	------------------------------------

## Returns

A pointer to the new first card, if the very first card removed

**delete\_First\_Card()**

```
CARD * delete_First_Card (  
    CARD * head)
```

Delete the very first card from the system.

**Parameters**

<i>head</i>	The pointer to the existing first card
-------------	--

**Returns**

A pointer to the new first card

**delete\_Intermediate\_or\_Last\_Card()**

```
CARD * delete_Intermediate_or_Last_Card (
    CARD * head)
```

Delete any of the existing card from the system without the first one.

**Parameters**

<i>head</i>	The pointer to the very first card
-------------	------------------------------------

**Returns**

A pointer to the first card

## 4.3 cards.h

[Go to the documentation of this file.](#)

```
00001 #ifndef CARDS_H_INCLUDED
00002 #define CARDS_H_INCLUDED
00003
00004
00010
00014 #define MAX_DESCRIPTION_LENGTH 250
00015
00019 #define MAX_NAME_LENGTH 30
00020
00024 #define DONE 1
00025
00029 #define NOT_DONE 0
00030 #include<stdbool.h>
00034 typedef struct CARD
00035 {
00036     char name[MAX_NAME_LENGTH];
00037     char description[MAX_DESCRIPTION_LENGTH];
00038     struct CARD* next_card;
00039     struct CARD* prev_card;
00040 }CARD;
00041
00048 CARD* create_Card(char name[MAX_NAME_LENGTH],char description[MAX_DESCRIPTION_LENGTH]);
00049
00054 void add_a_Card(CARD* head);
00055
00061 CARD* delete_a_Card(CARD* head);
00062
00068 CARD* delete_First_Card(CARD* head);
00069
00075 CARD* delete_Intermediate_or_Last_Card(CARD* head);
00076
00081 bool checkList();
00082 #endif
```

## 4.4 lists.c File Reference

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <stdbool.h>
#include "lists.h"
#include "cards.h"
```

### Functions

- `LIST * create_List (char name[MAX_NAME_LENGTH])`  
*Creates a new list in the system.*
- `void add_a_List (LIST *head)`  
*Insert a new list after the existing one.*
- `LIST * delete_a_List (LIST *head)`  
*Delete any of the existing list from the system.*
- `LIST * delete_First_List (LIST *head)`  
*Delete the very first list from the system.*
- `LIST * delete_Intermediate_or_Last_List (LIST *head)`  
*Delete any of the existing list from the system without the first one.*

#### 4.4.1 Function Documentation

##### add\_a\_List()

```
void add_a_List (
    LIST * head)
```

Insert a new list after the existing one.

##### Parameters

<i>head</i>	The pointer to the very first card
-------------	------------------------------------

##### create\_List()

```
LIST * create_List (
    char name[MAX_NAME_LENGTH])
```

Creates a new list in the system.

##### Parameters

<i>name</i>	The unique name of the list
-------------	-----------------------------

##### Returns

A pointer to the new list

**delete\_a\_List()**

```
LIST * delete_a_List (  
    LIST * head)
```

Delete any of the existing list from the system.

## Parameters

<i>head</i>	The pointer to the very first list
-------------	------------------------------------

## Returns

A pointer to the new first list, if the very first list removed

**delete\_First\_List()**

```
LIST * delete_First_List (  
    LIST * head)
```

Delete the very first list from the system.

## Parameters

<i>head</i>	The pointer to the existing first list
-------------	--

## Returns

A pointer to the new first list

**delete\_Intermediate\_or\_Last\_List()**

```
LIST * delete_Intermediate_or_Last_List (  
    LIST * head)
```

Delete any of the existing list from the system without the first one.

## Parameters

<i>head</i>	The pointer to the very first list
-------------	------------------------------------

## Returns

A pointer to the first list

**4.5 lists.h File Reference**

Header for managing the lists in the Tello\_Project\_Management\_System.

```
#include "cards.h"
```

## Data Structures

- struct [LIST](#)

*Represents a List in Trello\_Project\_Management\_System.*

## Macros

- #define [MAX\\_NAME\\_LENGTH](#) 30

*Maximum number of characters allowed for writing name.*

## Typedefs

- typedef struct LIST [LIST](#)

*Represents a List in Trello\_Project\_Management\_System.*

## Functions

- [LIST \\*](#) [create\\_List](#) (char name[[MAX\\_NAME\\_LENGTH](#)])  
*Creates a new list in the system.*
- void [add\\_a\\_List](#) ([LIST \\*](#)head)  
*Insert a new list after the existing one.*
- [LIST \\*](#) [delete\\_a\\_List](#) ([LIST \\*](#)head)  
*Delete any of the existing list from the system.*
- [LIST \\*](#) [delete\\_First\\_List](#) ([LIST \\*](#)head)  
*Delete the very first list from the system.*
- [LIST \\*](#) [delete\\_Intermediate\\_or\\_Last\\_List](#) ([LIST \\*](#)head)  
*Delete any of the existing list from the system without the first one.*

### 4.5.1 Detailed Description

Header for managing the lists in the Tello\_Project\_Management\_System.

Author

Data\_Structures\_Project\_Group\_01

### 4.5.2 Macro Definition Documentation

#### **MAX\_NAME\_LENGTH**

```
#define MAX_NAME_LENGTH 30
```

Maximum number of characters allowed for writing name.



### 4.5.3 Typedef Documentation

#### LIST

```
typedef struct LIST LIST
```

Represents a List in Trello\_Project\_Management\_System.

### 4.5.4 Function Documentation

#### add\_a\_List()

```
void add_a_List (
    LIST * head)
```

Insert a new list after the existing one.

##### Parameters

<i>head</i>	The pointer to the very first card
-------------	------------------------------------

#### create\_List()

```
LIST * create_List (
    char name[MAX_NAME_LENGTH])
```

Creates a new list in the system.

##### Parameters

<i>name</i>	The unique name of the list
-------------	-----------------------------

##### Returns

A pointer to the new list

#### delete\_a\_List()

```
LIST * delete_a_List (
    LIST * head)
```

Delete any of the existing list from the system.

##### Parameters

<i>head</i>	The pointer to the very first list
-------------	------------------------------------

##### Returns

A pointer to the new first list, if the very first list removed

#### delete\_First\_List()

```
LIST * delete_First_List (
    LIST * head)
```

Delete the very first list from the system.

**Parameters**

<i>head</i>	The pointer to the existing first list
-------------	--

**Returns**

A pointer to the new first list

**delete\_Intermediate\_or\_Last\_List()**

```
LIST * delete_Intermediate_or_Last_List (
    LIST * head)
```

Delete any of the existing list from the system without the first one.

**Parameters**

<i>head</i>	The pointer to the very first list
-------------	------------------------------------

**Returns**

A pointer to the first list

## 4.6 lists.h

[Go to the documentation of this file.](#)

```
00001 #ifndef LISTS_H_INCLUDED
00002 #define LISTS_H_INCLUDED
00003
00009
00010 #include "cards.h"
00011
00015 #define MAX_NAME_LENGTH 30
00016
00020 typedef struct LIST
00021 {
00022     char name[MAX_NAME_LENGTH];
00023     CARD* CARD;
00024     struct LIST* next_List;
00025     struct LIST* prev_List;
00026 }LIST;
00027
00033 LIST* create_List(char name[MAX_NAME_LENGTH]);
00034
00039 void add_a_List(LIST* head);
00040
00046 LIST* delete_a_List(LIST* head);
00047
00053 LIST* delete_First_List(LIST* head);
00054
00060 LIST* delete_Intermediate_or_Last_List(LIST* head);
00061 #endif
```

## 4.7 search.c File Reference

```
#include <stdio.h>
#include <string.h>
#include "lists.h"
#include "cards.h"
#include "search.h"
```

## Functions

- void `search_Card` (`CARD` \*head, char desired\_Card[`MAX_NAME_LENGTH`])  
*Search the desired card from the system.*
- void `search_List` (`LIST` \*head, char desired\_List[`MAX_NAME_LENGTH`])  
*Search the desired list from the system.*

### 4.7.1 Function Documentation

#### `search_Card()`

```
void search_Card (
    CARD * head,
    char desired_Card[MAX_NAME_LENGTH])
```

Search the desired card from the system.

##### Parameters

<i>head</i>	The pointer to the very first card
<i>desired_List</i>	The name of the card to search

#### `search_List()`

```
void search_List (
    LIST * head,
    char desired_List[MAX_NAME_LENGTH])
```

Search the desired list from the system.

##### Parameters

<i>head</i>	The pointer to the very first list
<i>desired_List</i>	The name of the list to search

## 4.8 search.h File Reference

Header for searching the cards or lists in the Tello\_Project\_Management\_System.

```
#include "lists.h"
#include "cards.h"
```

## Functions

- void `search_List` (`LIST` \*head, char desired\_List[`MAX_NAME_LENGTH`])  
*Search the desired list from the system.*
- void `search_Card` (`CARD` \*head, char desired\_Card[`MAX_NAME_LENGTH`])  
*Search the desired card from the system.*

### 4.8.1 Detailed Description

Header for searching the cards or lists in the Tello\_Project\_Management\_System.

Author

Data\_Structures\_Project\_Group\_01

### 4.8.2 Function Documentation

#### search\_Card()

```
void search_Card (  
    CARD * head,  
    char desired_Card[MAX_NAME_LENGTH])
```

Search the desired card from the system.

##### Parameters

<i>head</i>	The pointer to the very first card
<i>desired_List</i>	The name of the card to search

#### search\_List()

```
void search_List (  
    LIST * head,  
    char desired_List[MAX_NAME_LENGTH])
```

Search the desired list from the system.

##### Parameters

<i>head</i>	The pointer to the very first list
<i>desired_List</i>	The name of the list to search

## 4.9 search.h

[Go to the documentation of this file.](#)

```
00001 #ifndef SEARCH_H_INCLUDED  
00002 #define SEARCH_H_INCLUDED  
00003  
00009  
00010 #include"lists.h"  
00011 #include"cards.h"  
00012  
00018 void search_List(LIST* head,char desired_List[MAX_NAME_LENGTH]);  
00019  
00025 void search_Card(CARD* head, char desired_Card[MAX_NAME_LENGTH]);  
00026 #endif
```

## 4.10 user\_entry.c File Reference

```
#include <stdio.h>
#include <string.h>
#include "user_entry.h"
```

### Functions

- void [registration](#) ()  
*Store the information of the user for further Log In.*
- int [log\\_In](#) ()  
*Compare the stored information of the user from file to provide access.*

#### 4.10.1 Function Documentation

##### log\_In()

```
int log_In ()
```

Compare the stored information of the user from file to provide access.

##### registration()

```
void registration ()
```

Store the information of the user for further Log In.

## 4.11 user\_entry.h File Reference

Header for managing user entry.

### Data Structures

- struct [USER](#)  
*Represents a identity of a user.*

### Macros

- #define [MAX\\_LENGTH](#) 16  
*Maximum number of characters allowed for name and email-id.*

### Typedefs

- typedef struct USER [USER](#)  
*Represents a identity of a user.*

## Functions

- void [registration](#) ()  
*Store the information of the user for further Log In.*
- int [log\\_In](#) ()  
*Compare the stored information of the user from file to provide access.*

### 4.11.1 Detailed Description

Header for managing user entry.

Author

Data\_Structures\_Project\_Group\_01

### 4.11.2 Macro Definition Documentation

#### MAX\_LENGTH

```
#define MAX_LENGTH 16
```

Maximum number of characters allowed for name and email-id.

### 4.11.3 Typedef Documentation

#### USER

```
typedef struct USER USER
```

Represents a identity of a user.

### 4.11.4 Function Documentation

#### log\_In()

```
int log_In ()
```

Compare the stored information of the user from file to provide access.

#### registration()

```
void registration ()
```

Store the information of the user for further Log In.

## 4.12 user\_entry.h

[Go to the documentation of this file.](#)

```
00001 #ifndef USER_ENTRY_H_INCLUDED
00002 #define USER_ENTRY_H_INCLUDED
00003
00009
00013 #define MAX_LENGTH 16
00014
00018 typedef struct USER
00019 {
00020     char name[MAX_LENGTH];
00021     char email_id[MAX_LENGTH];
00022 }USER;
00023
00027 void registration();
00028
00032 int log_In();
00033 #endif
```

## Index

- add\_a\_Card
  - cards.c, [5](#)
  - cards.h, [8](#)
- add\_a\_List
  - lists.c, [11](#)
  - lists.h, [15](#)
- CARD, [2](#)
  - cards.h, [8](#)
  - description, [2](#)
  - LIST, [3](#)
  - name, [2](#)
  - next\_card, [2](#)
  - prev\_card, [2](#)
- cards.c, [4](#)
  - add\_a\_Card, [5](#)
  - checkList, [5](#)
  - create\_Card, [5](#)
  - delete\_a\_Card, [6](#)
  - delete\_First\_Card, [6](#)
  - delete\_Intermediate\_or\_Last\_Card, [6](#)
- cards.h, [7](#)
  - add\_a\_Card, [8](#)
  - CARD, [8](#)
  - checkList, [9](#)
  - create\_Card, [9](#)
  - delete\_a\_Card, [9](#)
  - delete\_First\_Card, [9](#)
  - delete\_Intermediate\_or\_Last\_Card, [10](#)
  - DONE, [8](#)
  - MAX\_DESCRIPTION\_LENGTH, [8](#)
  - MAX\_NAME\_LENGTH, [8](#)
  - NOT\_DONE, [8](#)
- checkList
  - cards.c, [5](#)
  - cards.h, [9](#)
- create\_Card
  - cards.c, [5](#)
  - cards.h, [9](#)
- create\_List
  - lists.c, [11](#)
  - lists.h, [15](#)
- delete\_a\_Card
  - cards.c, [6](#)
  - cards.h, [9](#)
- delete\_a\_List
  - lists.c, [11](#)
  - lists.h, [15](#)
- delete\_First\_Card
  - cards.c, [6](#)
  - cards.h, [9](#)
- delete\_First\_List
  - lists.c, [13](#)
  - lists.h, [15](#)
- delete\_Intermediate\_or\_Last\_Card
  - cards.c, [6](#)
  - cards.h, [10](#)
- delete\_Intermediate\_or\_Last\_List
  - lists.c, [13](#)
  - lists.h, [16](#)
- description
  - CARD, [2](#)
- DONE
  - cards.h, [8](#)
- email\_id
  - USER, [4](#)
- LIST, [3](#)
  - CARD, [3](#)
  - lists.h, [15](#)
  - name, [3](#)
  - next\_List, [3](#)
  - prev\_List, [3](#)
- lists.c, [11](#)
  - add\_a\_List, [11](#)
  - create\_List, [11](#)
  - delete\_a\_List, [11](#)
  - delete\_First\_List, [13](#)
  - delete\_Intermediate\_or\_Last\_List, [13](#)
- lists.h, [13](#)
  - add\_a\_List, [15](#)
  - create\_List, [15](#)
  - delete\_a\_List, [15](#)
  - delete\_First\_List, [15](#)
  - delete\_Intermediate\_or\_Last\_List, [16](#)
  - LIST, [15](#)
  - MAX\_NAME\_LENGTH, [14](#)
- log\_In
  - user\_entry.c, [19](#)
  - user\_entry.h, [20](#)
- MAX\_DESCRIPTION\_LENGTH
  - cards.h, [8](#)
- MAX\_LENGTH
  - user\_entry.h, [20](#)
- MAX\_NAME\_LENGTH
  - cards.h, [8](#)
  - lists.h, [14](#)
- name
  - CARD, [2](#)
  - LIST, [3](#)
  - USER, [4](#)
- next\_card
  - CARD, [2](#)
- next\_List
  - LIST, [3](#)
- NOT\_DONE
  - cards.h, [8](#)
- prev\_card
  - cards.c, [6](#)
  - cards.h, [10](#)

- CARD, [2](#)
- prev\_List
  - LIST, [3](#)
- registration
  - user\_entry.c, [19](#)
  - user\_entry.h, [20](#)
- search.c, [16](#)
  - search\_Card, [17](#)
  - search\_List, [17](#)
- search.h, [17](#)
  - search\_Card, [18](#)
  - search\_List, [18](#)
- search\_Card
  - search.c, [17](#)
  - search.h, [18](#)
- search\_List
  - search.c, [17](#)
  - search.h, [18](#)
- USER, [4](#)
  - email\_id, [4](#)
  - name, [4](#)
  - user\_entry.h, [20](#)
- user\_entry.c, [19](#)
  - log\_In, [19](#)
  - registration, [19](#)
- user\_entry.h, [19](#)
  - log\_In, [20](#)
  - MAX\_LENGTH, [20](#)
  - registration, [20](#)
  - USER, [20](#)