



Python Habit Tracker Application



Functionality

Screenshots showing the
functionality of the application
with example actions

Terminal Arguments

Account Creation / Login

Creation of a new account via terminal

```
PS C:\Users\Arbeit\HabitTracker> python cli.py --create_account
Please select your username: janedoe
Please select your password:
Account created successfully!
```

Login via terminal (error is due to the missing action as the user is supposed to specify one immediately)

```
PS C:\Users\Arbeit\HabitTracker> python cli.py --login --username janedoe --password janespassword
Login successful! Welcome, user 1!
Invalid action. Please specify a valid argument.
```

Login with specified action

Login and display all habits (sample user and sample habits that come with the application)

```
Login successful! Welcome, user 1!  
Invalid action. Please specify a valid argument.  
PS C:\Users\Arbeit\HabitTracker> python cli.py --login --username janedoe --password janespassword --display_habi  
Login successful! Welcome, user 1!  
Your habits:  
ID: 1| Name: Exercise | Periodicity: daily  
ID: 2| Name: Reading | Periodicity: weekly  
ID: 3| Name: Drink Water | Periodicity: daily  
ID: 4| Name: Meditation | Periodicity: weekly
```

Completion of a habit

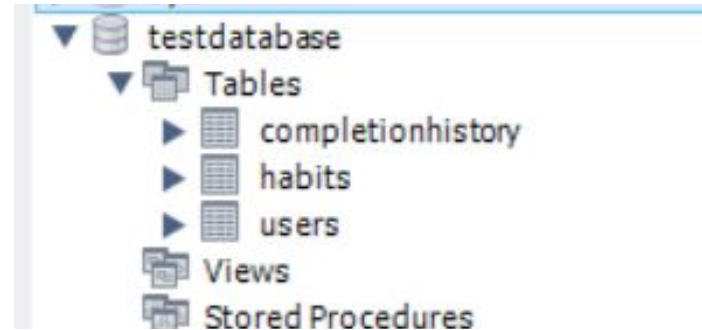
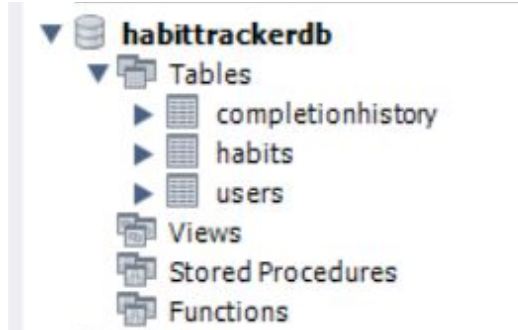
```
ID: 5| Name: Finances | Periodicity: monthly  
PS C:\Users\Arbeit\HabitTracker> python cli.py --login --username janedoe --password janespassword --complete_habit  
Login successful! Welcome, user 1!  
Your habits:  
ID : 1 | Name: Exercise | Periodicity: daily  
ID : 2 | Name: Reading | Periodicity: weekly  
ID : 3 | Name: Drink Water | Periodicity: daily  
ID : 4 | Name: Meditation | Periodicity: weekly  
ID : 5 | Name: Finances | Periodicity: monthly  
Please enter the ID of the habit you want to mark as complete: 1  
Longest streak for 'Exercise': 1 completions in a row.  
Current Streak for 'Exercise': 1 completions in a row.  
Habit 'Exercise' marked as complete '2025-01-07 11:25:23.335667'.  
Habit 'Exercise' marked as complete!
```

MySQL Workbench

Screenshots showing the
database and data that comes
with it

MySQL Database

Database setup (usage, and testing database)



identical structure, but only habittrackerdb comes with pre-installed data, as the pytest test suite for testdatabase automatically wipes all data from the database after every test unit

Pre-installed Habits & CompletionHistory

	habitID	habitname	periodicity	created_at	userrefID
▶	1	Exercise	daily	2025-01-07 11:00:27	1
	2	Reading	weekly	2025-01-07 11:00:48	1
	3	Drink Water	daily	2025-01-07 11:00:59	1
	4	Meditation	weekly	2025-01-07 11:01:08	1
	5	Finances	monthly	2025-01-07 11:01:27	1
•	NULL	NULL	NULL	NULL	NULL

5 habits dating of all three possible periodicities are defined for user1 (the example user by the name of janedoe)

	completionhistoryID	habitrefID	completedate
	64	4	2025-01-02 00:00:00
	65	5	2024-10-04 00:00:00
	66	5	2024-11-03 00:00:00
	67	5	2024-12-02 00:00:00
	68	5	2025-01-04 00:00:00
•	NULL	NULL	NULL

completionhistory2 ▾

68 completion history entries are defined for all pre-installed habits dating back anywhere from 4 to 10 weeks

Notes on Database Installation

```
5
6 #Creation of a new database
7 db = mysql.connector.connect(
8     host = "localhost",
9     user = "root",
10    passwd = "testingpassword",
11    database = "habitrackerdb"
12    #database = "testdatabase" #Uncomment this for pytest tests, use "habitrackerdb" for real use
13 )
14
```

This database connection in database.py must be set up correctly. The host, user, and password (lines 8 - 10) are personalized by the user during the installation, establishing their personal connection.

For this, MySQL Server, MySQL Shell, and MySQL Workbench must be installed (installation of requirements.txt ensures this)

Run using
test_program.py file

Pytest test suite run on the
duplicated database

Pytest

Pytest Result

Database connection must be set up correctly before running pytest or else all data will be deleted!

Pytest can be run in the terminal as seen in the screenshot below (pytest test_program.py).

If everything was installed correctly, the following output is returned:

```
===== 14 passed in 1.01s =====
PS C:\Users\Arbeit\HabitTracker> pytest test_program.py
===== test session starts =====
platform win32 -- Python 3.11.2, pytest-8.3.4, pluggy-1.5.0
rootdir: C:\Users\Arbeit\HabitTracker
collected 14 items

test_program.py ..... [100%]

===== 14 passed in 1.10s =====
PS C:\Users\Arbeit\HabitTracker> 
```

Adjustments pre-run

```
5
6 #Creation of a new database
7 db = mysql.connector.connect(
8     host = "localhost",
9     user = "root",
10    passwd = "testingpassword",
11    database = "habitrackerdb"
12    #database = "testdatabase" #Uncomment this for pytest tests, use "habitrackerdb" for real use
13 )
14
```

Before being able to run the test suite, the user must comment the name of the database (line 11) and uncomment the testdatabase (line 12). As the test suite wipes the database clean, this prevents the pre-installed data to be deleted. Before the user can use the database, they must ensure that the database connection is reverted back to this original state in order to use to correct database.