```python
import time
import pandas as pd
import numpy as np

CITY_DATA = { 'chicago': 'chicago.csv',
              'new_york_city': 'new_york_city.csv',
              'washington': 'washington.csv' }

def get_filters():
    """
    Asks user to specify a city, month, and day to analyze.

    Returns:
        (str) city - name of the city to analyze
        (str) month - name of the month to filter by, or "all" to apply no month filter
        (str) day - name of the day of week to filter by, or "all" to apply no day filter
    """
    print('Hello! Let\'s explore some US bikeshare data!')
    # get user input for city (chicago, new york city, washington). HINT: Use a while loop to handle invalid inputs
    city = input('ENTER THE CITY (chicago, new_york_city OR washington): ')
    while city not in ['chicago', 'new_york_city', 'washington']:
        city = input ("CHOOSE BETWEEN chicago, new_york_city OR washington: ").lower()

    # get user input for month (all, january, february, ... , june)
    month = input('ENTER MONTH (all, january, february, ... , june): ').lower()
    while month not in ['all','january', 'february', 'march', 'april', 'may', 'june']:
        month = input('ENTER MONTH (all, january, february, ... , june) : ').lower()


    # get user input for day of week (all, monday, tuesday, ... sunday)
    day = input('ENTER DAY (all, monday, tuesday, ... sunday) : ').lower()
    while day not in ['all','monday', 'tuesday', 'wednesday', 'thursday', 'friday', 'saturday', 'sunday']:
        day = input('ENTER DAY (all, monday, tuesday, ... sunday) : ').lower()

    print('-'*40)
    return city, month, day


def load_data(city, month, day):
    """
    Loads data for the specified city and filters by month and day if applicable.

    Args:
        (str) city - name of the city to analyze
        (str) month - name of the month to filter by, or "all" to apply no month filter
        (str) day - name of the day of week to filter by, or "all" to apply no day filter
    Returns:
        df - Pandas DataFrame containing city data filtered by month and day
    """
    df = pd.read_csv('{}.csv'.format(city))
    df['Start Time'] = pd.to_datetime(df['Start Time'])
    df['End Time'] = pd.to_datetime(df['End Time'])
    df['month'] = df['Start Time'].dt.month
    if month != 'all':
        months = ['january', 'february', 'march', 'april', 'may', 'june']
        month = months.index(month) + 1
        df = df[df['month'] == month]
    df['day_of_week'] = df['Start Time'].dt.day_name()
    if day != 'all':
        df = df[df['day_of_week'] == day.title()]
    return df


```

```python
def time_stats(df):
    """Displays statistics on the most frequent times of travel."""

    print('\nCalculating The Most Frequent Times of Travel...\n')
    start_time = time.time()

    # display the most common month
    print("The most common month is: ", df['month'].value_counts().idxmax())

    # display the most common day of week
    print("The most common day is: ", df['day_of_week'].value_counts().idxmax())

    # display the most common start hour
    df['hour'] = df['Start Time'].dt.hour
    print("The most common hour is: ", df['hour'].value_counts().idxmax())


    print("\nThis took %s seconds." % (time.time() - start_time))
    print('-'*40)


def station_stats(df):
    """Displays statistics on the most popular stations and trip."""

    print('\nCalculating The Most Popular Stations and Trip...\n')
    start_time = time.time()

    # display most commonly used start station
    print("The most common start station is: ", df ['Start Station'].value_counts().idxmax())

    # display most commonly used end station
    print("The most common end station is: ", df['End Station'].value_counts().idxmax())

    # display most frequent combination of start station and end station trip
    print("The most frequent combination of start station and end station trip")
    most_common_start_and_end_stations = df.groupby(['Start Station', 'End Station']).size().nlargest(1)
    print(most_common_start_and_end_stations)

    print("\nThis took %s seconds." % (time.time() - start_time))
    print('-'*40)


def trip_duration_stats(df):
    """Displays statistics on the total and average trip duration."""

    print('\nCalculating Trip Duration...\n')
    start_time = time.time()

    # display total travel time
    total_duration = df['Trip Duration'].sum() / 3600.0
    print("total travel time in hours is: ", total_duration)

    # display mean travel time
    mean_duration = df['Trip Duration'].mean() / 3600.0
    print("mean travel time in hours is: ", mean_duration)


    print("\nThis took %s seconds." % (time.time() - start_time))
    print('-'*40)


def user_stats(df):
    """Displays statistics on bikeshare users."""

    print('\nCalculating User Stats...\n')
    start_time = time.time()

    # Display counts of user types
    user_types = df['User Type'].value_counts()
    print(user_types)

    # Display counts of gender
    try:
        user_gender = df['Gender'].value_counts()
        print("Gender is:", user_gender)


        # Display earliest, most recent, and most common year of birth
        earliest_year_of_birth = int(df['Birth Year'].min())
        most_recent_year_of_birth = int(df['Birth Year'].max())
        most_common_year_of_birth = int(df['Birth Year'].value_counts().idxmax())
        print("The earliest year of birth is:",earliest_year_of_birth,
          ", most recent one is:",most_recent_year_of_birth,
           "and the most common one is: ",most_common_year_of_birth)
    except:
        print('No filter with gender allowed in Washington.')
    print("\nThis took %s seconds." % (time.time() - start_time))
    print('-'*40)

def display_data(df):
    more_data = input("Would you like to view 5 rows of individual trip data? Enter yes or no? ").lower()
    start_loc = 0
    while more_data == 'yes':
        print(df.iloc[start_loc:start_loc+5])
        start_loc += 5
        more_data = input("Do you wish to continue? Enter yes or no? ").lower()

    return df

def main():
    while True:
        city, month, day = get_filters()
        df = load_data(city, month, day)

        time_stats(df)
        station_stats(df)
        trip_duration_stats(df)
        user_stats(df)
        display_data(df)

        restart = input('\nWould you like to restart? Enter yes or no.\n')
        if restart.lower() != 'yes':
            break


if __name__ == "__main__":
    main()

```