Experiment 3: Implementation of Multi-Armed Bandits using Open GYM AI API

```python
import warnings

n_arms = env.action_space.n
num_episodes = 500

epsilon = 0.1

q_values = np.zeros(n_arms)
counts = np.zeros(n_arms)

all_rewards = []
action_history = []

warnings.filterwarnings("ignore", category=DeprecationWarning)
```

```python
[7] import gym
    import gym_bandits
    import numpy as np
```

```python
[9] env=gym.make("MultiarmedBandits-v0")
```

```python
[10] print(env.observation_space)
```
```
Discrete(1)
```

```python
[11] print(env.action_space)
```
```
Discrete(10)
```

```python
reward_sums = np.zeros(n_arms)

for episode in range(num_episodes):
    env.reset()

    if np.random .rand() <epsilon:
        action = env.action_space.sample()
    else:
        action = np.argmax(q_values)

    observation, reward, done, info = env.step(action)
    counts[action] += 1
    reward_sums[action] += reward

    q_values[action] = reward_sums[action] / counts[action]

    all_rewards.append(reward)
    action_history.append(action)

env.close()

print("Estimated Q-values for each arm :" , np.round(q_values,2))
print("Number of times each arm was selected:", counts.astype((int)))
print("Average reward over all episodes:", round(np.mean(all_rewards), 3 ))

plt.figure(figsize=(14,5))

plt.subplot(1,2,1)
plt.plot(np.cumsum(all_rewards), color='blue')
plt.xlabel("Episode")
plt.ylabel("Cumulative Reward")
plt.title("Cumulative Reward Over Episodes")

plt.subplot(1,2,2)
plt.bar(np.arange(n_arms), counts, color='green')
plt.xlabel("Arm Index")
plt.ylabel("Times Selected")
plt.title("Action Selection Frequency")

plt.tight_layout()
plt.show()
```

```
Estimated Q-values for each arm : [-0.16  0.08 -0.08 -0.37 -0.46 -0.37 -0.24 -0.12 -0.08 -0.01]
Number of times each arm was selected: [ 21 176  61   8  11  20  14   4  39 146]
Average reward over all episodes: -0.039
```