# Part 2 HMM Assignment

## Capstone Project: Low-Bandwidth Educational Content Platform with Offline Support

**Project Idea:** My mission is to improve the delivery of educational resources to children in rural areas by developing a low-bandwidth educational content platform using HTML, CSS, JavaScript, and Node.js, employing my full-stack web-development skills. This "Web-Based Quiz and E-Learning Platform with Offline Support" will allow children to study even with poor internet connection, aligns with SDG 4 (Quality Education), and showcases web-development prowess.

## Hidden Markov Model (HMM) Application

A Hidden Markov Model (HMM) could be used in your project to **model and predict a student's underlying knowledge state** (the hidden states) based on their observable interactions with the platform, such as quiz answers and engagement with learning materials. This could enhance the platform's adaptive learning capabilities, even in low-bandwidth environments by pre-caching relevant content based on predicted knowledge.

**1. Describe the Observations:** The measurable data (observations) that the HMM would use are the observable actions and responses of the students on the platform. These could include:

- **Quiz Answers:** Correct or incorrect responses to multiple-choice questions, fill-in-the-blank, or short-answer questions.
- **Time Taken:** The time a student spends answering a question or engaging with a learning module.
- **Attempt Count:** The number of attempts made on a particular question before getting it correct.
- **Navigation Patterns:** The sequence of learning modules or topics a student accesses.
- **Engagement Metrics:** Whether a student completes a module, watches a video to completion, or skips sections.

For simplification, a common approach for educational HMMs is to primarily use **sequences of correct/incorrect answers on quiz questions** as the observations. For example, a sequence like "Correct, Correct, Incorrect, Correct" for a series of questions.

**2. Type of HMM Problem:** If you don’t know the hidden states (the student's exact knowledge level or mastery of a concept) in advance, this is a **Learning (or Training) Problem**.

- **Reasoning:** In this scenario, you wouldn't initially have labeled data explicitly stating a student's "mastered," "partially mastered," or "not mastered" state for each concept

at every point in time. Instead, you would observe their sequence of quiz answers and interactions, and the HMM would learn the probabilities that govern these hidden knowledge states and their transitions, as well as the probabilities of observing specific answers given a certain hidden state.

**3. Training Algorithm:**

The most common algorithm for the HMM Learning Problem where hidden states are unknown is the **Baum-Welch algorithm**. This is an Expectation-Maximization (EM) algorithm.

**a. What values are known at the start?**

- **Observation Sequences (O):** The sequences of observable student actions/responses (e.g., quiz answer sequences) collected from various students.
- **Number of Hidden States (N):** You would need to pre-define the number of possible hidden knowledge states. For example, you might define 2 states (e.g., "Knowledgeable," "Not Knowledgeable") or 3 states (e.g., "Novice," "Intermediate," "Mastered"). This number is typically chosen based on domain expertise or experimentation.
- **Number of Unique Observations (M):** The size of your observation alphabet (e.g., if observations are just "Correct" and "Incorrect," then M=2).
- **Initial Estimates for HMM Parameters:** The Baum-Welch algorithm requires initial (often random) estimates for the HMM parameters (initial state probabilities, state transition probabilities, and observation emission probabilities). These initial estimates are then iteratively refined.

**b. What values are unknown and need to be learned?** The values that are unknown and need to be learned by the Baum-Welch algorithm are the core HMM parameters, often collectively referred to as $\lambda=(A,B,\pi)$:

- **State Transition Probability Matrix (A):** The probabilities of transitioning from one hidden knowledge state to another (e.g., the probability of a student going from "Not Knowledgeable" to "Knowledgeable" after reviewing a module).
- **Observation (Emission) Probability Matrix (B):** The probabilities of observing a particular output (e.g., "Correct Answer") given that the student is in a specific hidden knowledge state (e.g., the probability of answering correctly if in the "Not Knowledgeable" state).
- **Initial State Probabilities (π):** The probabilities of starting in each hidden knowledge state (e.g., the probability that a new student begins in the "Novice" state).

**c. Parameter Updates: Which HMM parameters will your training algorithm update?** The Baum-Welch training algorithm iteratively updates all three core HMM parameters:

- **Initial State Probabilities ($\pi i$):** Updated based on the expected number of times each hidden state is observed at the beginning of an observation sequence.
- **State Transition Probabilities (aij):** Updated based on the expected number of transitions from hidden state i to hidden state j across all observation sequences.

- **Observation (Emission) Probabilities (bj(k)):** Updated based on the expected number of times observation k is emitted from hidden state j across all observation sequences.