Quiz/Survey Application

# Step 1: Define the Problem and Goal

1. Identify the Problem
   1. What is the purpose of the Quiz/Survey Application? Educate
2. Define the Goal
   1. What do we want to achieve with the application?
      1. Create a fun quizzing process
      2. Gamify
      3. Spaced Repetition
   2. Create your own Quiz/Survey/Flashcards

# Step 2: Identify the Target Audience

1. Target Audience:
   1. Who is the target audience? (me, students, professionals)
2. Understand Their Needs:
   1. What are their needs, preferences, and pain points?
   2. What are their technical skills and familiarity with online applications?

# Step 3: Determine the Features and Functionality

1. Brainstorm Features:
   1. Quiz/Survey Creation and Management
   2. User Authentication and Authorization
   3. Various question types (multiple-choice, true/false, open-ending)
   4. Scoring and feedback mechanisms
   5. User profiles and progress tracking and gamification
2. Prioritize Features:
   1. Based on importance and feasibility.

# Step 4: Define the Technical Requirements

1. Front-end: HTML, CSS, JavaScript, HTMX
2. Backend: PHP

# Step 5: Create a Wireframe and User Flow

1. Wireframe:
   1. Drawing an example
   2. Identifying the elements, buttons, forms, menus, …
2. User Flow Diagram:
   1. Show the sequence of steps for a user to complete a quiz or survey
   2. What any pain points or areas of improvement

# Step 6: Plan the Data Model and Storage

1. Define the Data Model
   1. Identify the entities (Users, Quizzes, Question, Answers, etc)
   2. Define the Relationships between the entities
2. Determine Storage Requirements
   1. Choose a database scheme and data storage (mySql)
   2. Consider data security and backup strategies

# Step 7: Estimate the Resources and Timeline

1. Estimate Resources:
   1. Personnel: Developers, designers, and project managers
   2. Time: Development, Testing, and Deployment
   3. Budget: Hardware, software, and infrastructure costs.
2. Create a Timeline:
   1. Break down the project into smaller tasks and milestones.
   2. Identify critical dependencies and potential roadblocks

# Step 8: Plan the Testing and Quality Assurance

1. Testing Strategies:
   1. Unit testing, integration testing, and user acceptance testing
   2. Automated testing and manual testing
2. Quality Assurance Process:
   1. Identify the quality metrics and standards
   2. Establish a testing schedule and timeline

# Step 9: Plan the Deployment and Maintenance

1. Deployment Strategy:
   1. Cloud hosting, Virtual private server, or dedicated server?
   2. Load balancing and scalability considerations
2. Maintenance and Updates:
   1. Regular backups and data recovery
   2. Bug fixes and feature updates
   3. Monitoring and analytics for performance and user behavior.