Aman Adams

ST10290748

Information Systems 3D

ICE Task 1

Section A:

1) The main purpose of CAPTCHA is to differentiate between human users and automated bots, by requiring a task that's easy for humans but hard for automated scripts to solve, CAPTCHA prevents spam and fake account creation.

2) CAPTCHA prevents bot submissions by presenting challenges that require human-like perception (e.g., image recognition, pattern matching) or logic (e.g., simple puzzles) that most automated bots cannot solve easily. This prevents bots from automatically submitting forms without manual human interaction.

3) Text-based CAPTCHAs – distorted letters/numbers that must be typed in.
   Image-based CAPTCHAs – selecting specific images (e.g., "Click all the photos with a motorbike in them").
   Math/problem-solving CAPTCHAs – solve a small arithmetic problem.

4) Accessibility issues – visually impaired, dyslexic or neurodivergent users may struggle with distorted text or image recognition.
   User frustration – solving CAPTCHAs can be time-consuming and irritating, especially if it's hard to read or requires multiple attempts. If the user gets it wrong, despite being human, it could lead to being unable to continue and being frustrated.

5) No, CAPTCHAs can help reduce automated submissions however sophisticated bots can still use machine learning or human CAPTCHA farms to bypass them. They do not protect against manual spam, SQL injections, or other web vulnerabilities. You still need multi-layered security (e.g. rate-limiting, IP blocking, email verification, and strong input validation.)

Section B:

1) SQL Injection vulnerability.

2) An example of a SQL comment operator is "- - ", so everything after it is ignored. This means the password check is completely skipped.

3) An attacker logs in as the admin account without knowing the password, they then gain full access to the system leaking important private data.

4) Hash and verify passwords -- Never store plain-text passwords. Use “password_hash()” and “password_verify()” in PHP. This way, even if data is leaked, attackers cannot directly use the passwords.

   Use prepared statements (parameterized queries) -- Validate and sanitize all user input and apply least-privilege permissions to the database account.