COLLEGECODE:9528

COLLEGE NAME: SCAD COLLEGE OF ENGINEERING AND TECHNOLOGY

DEPARTMENT COMPUTER SCIENCE ENGINEERING

STUDENT NM ID: F99679D4DE022AAAFB54C276C36A6B1C

Roll no : 952823104164

DATE : 26.09.2025

Completed the project named as:

Phase 1

TECHNOLOGYPROJECTNAME : **USER REGISTERATION AND VALIDATION**

SUBMITTED By,

NAME: SUJITHRA N

MOBILE: 9944738661

Phase 3– MVP Implementation

1.Project Setup

The User Registration Validation project begins with a proper setup of the development environment. Python is selected because of its simplicity and wide availability of libraries for validation and database handling. The project structure is organized into different folders such as src for source code, tests for test cases, and db for database files. SQLite is configured as the local database since it is lightweight, portable, and requires no external server installation. All dependencies are installed using pip, and a requirements.txt file is created to ensure that the same environment can be reproduced easily. This phase also includes initializing a GitHub repository, setting up .gitignore to exclude unnecessary files, and documenting setup steps in a README.md file.

The setup begins with preparing the development environment using Python as the programming language. A clear folder structure is created for source code, test cases, and database files. SQLite is chosen as the database since it is lightweight and easy to integrate for rapid development. Dependencies are installed using pip, and an initial GitHub repository is created to track all progress and ensure collaborative development.

2. Core Features Implementation

The main features of this project include validating user information such as username, email address, password, and phone number at the time of registration. The system checks that the username is unique, the email follows a standard format, and the password is strong (minimum length, inclusion of numbers, and special characters). These rules prevent invalid or insecure data from being stored. Additionally, feedback messages are provided to guide users to correct errors in their input. This ensures that the system is both user-friendly and secure.

The core of the system is validating user inputs during registration. Four main fields are considered: username, email, password, and phone number. The username must be unique to avoid duplication in the database. Email addresses are checked with regular expressions to confirm that they follow the correct format (e.g., name@example.com). Passwords are validated to ensure strength; they must be at least eight characters long, include both numbers and letters, and contain at least one special character. This prevents weak passwords from being accepted. If validation fails, the user receives an appropriate error message such as “Invalid email format” or “Password too weak.” These rules make the registration process secure and reliable.

3. Data Storage (Local State / Database)

Once validation is passed, user details are saved in the database. At the MVP level, SQLite is used since it allows quick prototyping and works without an external server. The database schema includes fields for username, email, password, and phone. Before inserting a new record, the system queries the database to ensure that the username and email are not already registered. This check enforces uniqueness and prevents duplicate accounts. In this stage, passwords may be stored in plain text for testing purposes, but the design considers adding hashing techniques (such as SHA-256 or bcrypt) in later phases to ensure security. Using SQLite allows the team to focus on functionality without worrying about complex server configurations.

```
// Install first: npm install express sqlite3 body-parser cors
const express = require("express");
const sqlite3 = require("sqlite3").verbose();
const bodyParser = require("body-parser");
const cors = require("cors");

const app = express();
app.use(bodyParser.json());
app.use(cors());

// Database setup
const db = new sqlite3.Database("./users.db", (err) => {
  if (err) console.error(err.message);
  console.log("Connected to SQLite database.");
});

db.run(`CREATE TABLE IF NOT EXISTS users (
  id INTEGER PRIMARY KEY AUTOINCREMENT,
  username TEXT UNIQUE,
  email TEXT UNIQUE,
  password TEXT,
  phone TEXT
)`);

// Validation function
function validateUser(user) {
  if (user.username.length < 4) return "Username must be at least 4
characters.";
  const emailRegex = /^[^ ]+@[^ ]+\.[a-z]{2,3}$/;
```