Code

```
#include <lpc214x.h>

// Delay function
void delay_ms(unsigned int ms) {
    unsigned int i, j;
    for (i = 0; i < ms; i++)
        for (j = 0; j < 6000; j++);
}

// LCD Functions
#define LCD_DIR IO1DIR
#define LCD_OUT IO1SET
#define LCD_CLR IO1CLR

#define RS (1 << 16)
#define EN (1 << 17)
#define D4 (1 << 18)
#define D5 (1 << 19)
#define D6 (1 << 20)
#define D7 (1 << 21)

void lcd_pulse() {
    LCD_OUT |= EN;
```

```
    delay_ms(1);
    LCD_CLR |= EN;
    delay_ms(1);
}

void lcd_send4(unsigned char data) {
    LCD_CLR |= D4 | D5 | D6 | D7;
    if (data & 0x01) LCD_OUT |= D4;
    if (data & 0x02) LCD_OUT |= D5;
    if (data & 0x04) LCD_OUT |= D6;
    if (data & 0x08) LCD_OUT |= D7;
    lcd_pulse();
}

void lcd_cmd(unsigned char cmd) {
    LCD_CLR |= RS;
    lcd_send4(cmd >> 4);
    lcd_send4(cmd & 0x0F);
    delay_ms(2);
}

void lcd_data(unsigned char data) {
    LCD_OUT |= RS;
    lcd_send4(data >> 4);
    lcd_send4(data & 0x0F);
    delay_ms(2);
}
```

```
void lcd_init() {
    LCD_DIR |= RS | EN | D4 | D5 | D6 | D7;
    delay_ms(20);
    lcd_cmd(0x02);  // 4-bit
    lcd_cmd(0x28);  // 2 line, 5x7
    lcd_cmd(0x0C);  // Display ON
    lcd_cmd(0x06);  // Auto increment
    lcd_cmd(0x01);  // Clear
}

void lcd_print(char *str) {
    while (*str)
        lcd_data(*str++);
}

void lcd_setCursor(unsigned int row, unsigned int col) {
    unsigned char pos = (row == 1) ? (0x80 + col) : (0xC0 + col);
    lcd_cmd(pos);
}

// Keypad functions
#define KEYPAD_ROWS 0x0003C000  // P0.14–P0.17 (Rows)
#define KEYPAD_COLS 0x00003C00  // P0.10–P0.13 (Cols)

void keypad_init() {
    IO0DIR |= KEYPAD_COLS;  // Columns as output
```

```
    IO0DIR &= ~KEYPAD_ROWS; // Rows as input
}

char keypad_getkey() {
    unsigned int row, col;
    char keys[4][4] = {
        {'7','8','9','/'},
        {'4','5','6','*'},
        {'1','2','3','-'},
        {'C','0','=','+'}
    };

    for (col = 0; col < 4; col++) {
        IO0SET = KEYPAD_COLS;
        IO0CLR = (1 << (10 + col)); // drive one column low
        delay_ms(2);

        for (row = 0; row < 4; row++) {
            if (!(IO0PIN & (1 << (14 + row)))) {
                while (!(IO0PIN & (1 << (14 + row)))); // wait for release
                return keys[row][col];
            }
        }
    }
    return 0;
}
```

```
// Buzzer and LED
#define BUZZER (1 << 18)
#define LED    (1 << 19)

void buzzer_beep() {
    IO0SET = BUZZER;
    delay_ms(200);
    IO0CLR = BUZZER;
}

void led_blink() {
    IO0SET = LED;
    delay_ms(300);
    IO0CLR = LED;
}

// Main logic
void token_service(char type) {
    static int token = 1;
    char buffer[16];

    lcd_cmd(0x01);
    lcd_setCursor(1, 0);
    lcd_print("Service: ");
    lcd_data(type);

    lcd_setCursor(2, 0);
```

```
    sprintf(buffer, "Token: %d", token++);
    lcd_print(buffer);

    buzzer_beep();
    led_blink();
    delay_ms(1000);
}

int main() {
    PINSEL0 = 0x00000000;
    PINSEL1 = 0x00000000;
    IO0DIR |= BUZZER | LED; // buzzer & LED output

    lcd_init();
    keypad_init();

    lcd_setCursor(1, 0);
    lcd_print("Welcome to Bank");

    while (1) {
        char key = keypad_getkey();
        if (key) {
            if (key == '1') token_service('A');  // Deposit
            else if (key == '2') token_service('B'); // Withdraw
            else if (key == '3') token_service('C'); // Account Open
            else if (key == '4') token_service('D'); // Loan
            else {
```

```
            lcd_cmd(0x01);
            lcd_print("Invalid Option");
            delay_ms(1000);
        }
    }
  }
}
```