1.
```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

/* ------------------- FILE NAMES ------------------- */
#define CREDENTIAL_FILE "credentials.txt"
#define SERVED_FILE     "served_customers.txt"

/* ------------------- QUEUE SETTINGS ------------------- */
#define MAX 50   // maximum customers in queue

/* ------------------- STRUCTURE ------------------- */
typedef struct
{
    int token;
    char name[100];
    char service[100];
} Customer;

/* ------------------- QUEUE STRUCT ------------------- */
typedef struct
{
    Customer arr[MAX];
    int front;
    int rear;
    int nextToken;
} CustomerQueue;

/* ------------------- QUEUE FUNCTIONS ------------------- */

void initQueue(CustomerQueue *q)
{
    q->front = -1;
    q->rear = -1;
    q->nextToken = 1;
}

int isFull(CustomerQueue *q)
{
    return (q->front == 0 && q->rear == MAX - 1) ||
        (q->front == q->rear + 1);
}

int isEmpty(CustomerQueue *q)
{
    return (q->front == -1);
}

void clearInputBuffer()
{
    int ch;
    while ((ch = getchar()) != '\n' && ch != EOF) {}
}

void removeNewline(char *str)
{
    size_t len = strlen(str);
    if (len > 0 && str[len - 1] == '\n')
        str[len - 1] = '\0';
}

void enqueueCustomer(CustomerQueue *q)
{
    if (isFull(q))
    {
        printf("\nQueue is FULL! Cannot add more customers.\n");
        return;
    }

    Customer c;
    c.token = q->nextToken++;

    printf("\nEnter Customer Name: ");
    clearInputBuffer();
    fgets(c.name, sizeof(c.name), stdin);
    removeNewline(c.name);

    printf("Enter Service Type (Deposit/Withdrawal/Enquiry etc.): ");
    fgets(c.service, sizeof(c.service), stdin);
    removeNewline(c.service);
    if (q->front == -1)
        q->front = 0;

    q->rear = (q->rear + 1) % MAX;
    q->arr[q->rear] = c;

    printf("\nCustomer Added to Queue.\n");
    printf("Assigned Token Number: %d\n", c.token);
}

void serveCustomer(CustomerQueue *q)
{
    if (isEmpty(q))
    {
        printf("\nNo customers in the queue.\n");
        return;
    }

    Customer c = q->arr[q->front];

    if (q->front == q->rear)
        q->front = q->rear = -1;
    else
        q->front = (q->front + 1) % MAX;

    printf("\nServing Customer:\n");
    printf("Token: %d | Name: %s | Service: %s\n", c.token, c.name, c.service);

    FILE *file = fopen(SERVED_FILE, "a");
    if (file != NULL)
    {
        fprintf(file, "Token: %d | Name: %s | Service: %s\n",
                c.token, c.name, c.service);
        fclose(file);
    }
}

void displayQueue(CustomerQueue *q)
{
    if (isEmpty(q))
    {
        printf("\nNo customers in the queue.\n");
        return;
    }

    printf("\n----- CURRENT QUEUE -----\n");
    int i = q->front;

    while (1)
    {
        printf("Token: %d | Name: %s | Service: %s\n",
               q->arr[i].token, q->arr[i].name, q->arr[i].service);

        if (i == q->rear)
            break;

        i = (i + 1) % MAX;
    }
}

void peekCustomer(CustomerQueue *q)
{
    if (isEmpty(q))
    {
        printf("\nNo customers in the queue.\n");
        return;
    }

    Customer c = q->arr[q->front];
    printf("\nNext Customer to be Served:\n");
    printf("Token: %d | Name: %s | Service: %s\n",
           c.token, c.name, c.service);
}

/* ------------------- LOGIN SYSTEM ------------------- */

int loginSystem()
{
    FILE *infile = fopen(CREDENTIAL_FILE, "r");

    char savedUser[50] = "admin";
    char savedPass[50] = "admin123";
    if (infile != NULL)
    {
        fscanf(infile, "%49s %49s", savedUser, savedPass);
        fclose(infile);
    }

    char user[50], pass[50];
    printf("\n======== LOGIN SCREEN ========\n");
    printf("Username: ");
    scanf("%49s", user);
    printf("Password: ");
    scanf("%49s", pass);

    if (strcmp(user, savedUser) == 0 && strcmp(pass, savedPass) == 0)
    {
        printf("\nLogin Successful!\n");
        return 1;
    }
    else
    {
        printf("\nInvalid Username or Password!\n");
        return 0;
    }
}

/* ------------------- MAIN FUNCTION ------------------- */

int main()
{
    if (!loginSystem())
    {
        printf("\nLogin Failed. Exiting...\n");
        return 0;
    }

    CustomerQueue q;
    initQueue(&q);

    int choice;

    do
    {
        printf("\n===== BANK QUEUE SIMULATION (C) =====\n");
        printf("1. New Customer Enters Queue (Enqueue)\n");
        printf("2. Serve Next Customer (Dequeue)\n");
        printf("3. Display Waiting Customers\n");
        printf("4. Show Next Customer to be Served (Front)\n");
        printf("5. Exit\n");
        printf("Enter choice: ");
        scanf("%d", &choice);

        switch (choice)
        {
        case 1:
            enqueueCustomer(&q);
            break;
        case 2:
            serveCustomer(&q);
            break;
        case 3:
            displayQueue(&q);
            break;
        case 4:
            peekCustomer(&q);
            break;
        case 5:
            printf("\nExiting Queue Simulation...\n");
            break;
        default:
            printf("\nInvalid Choice! Try Again.\n");
        }

    } while (choice != 5);

    return 0;
}
```