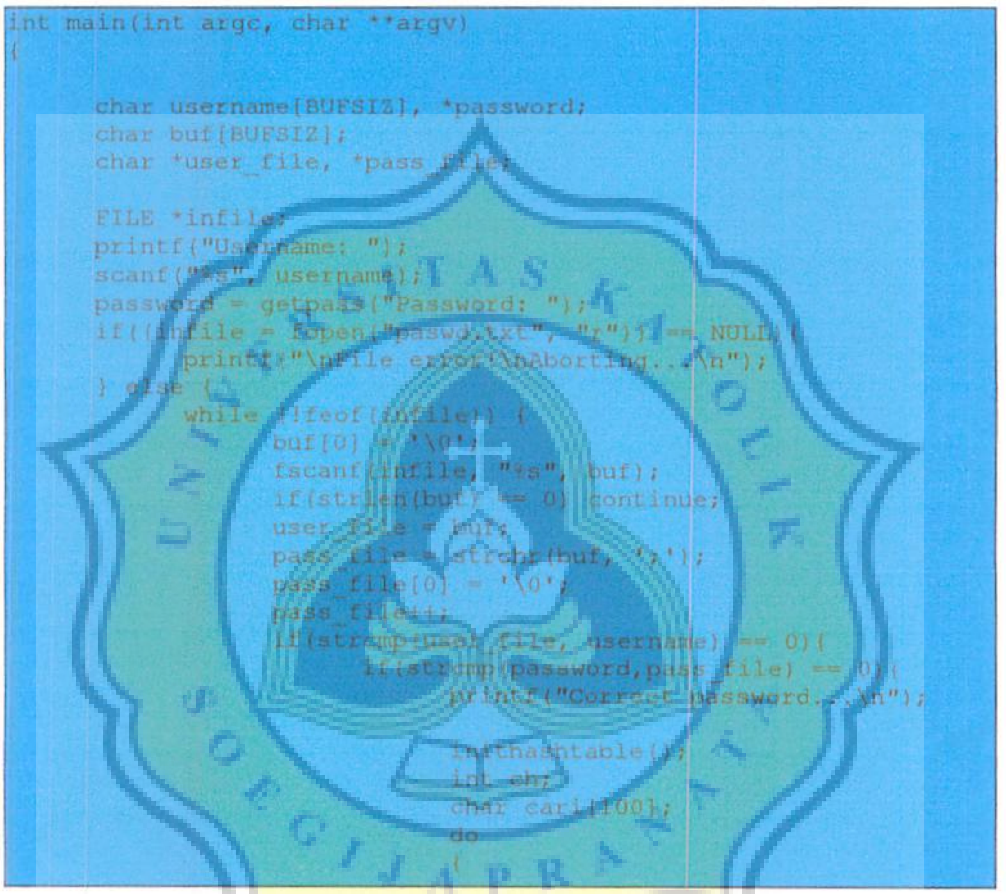
CHAPTER V

IMPLEMENTATION AND TESTING

5.1 Implementation

5.1 .1 The user should login username based on account number take from file txt and then insert the password if it is true then it will be correct.



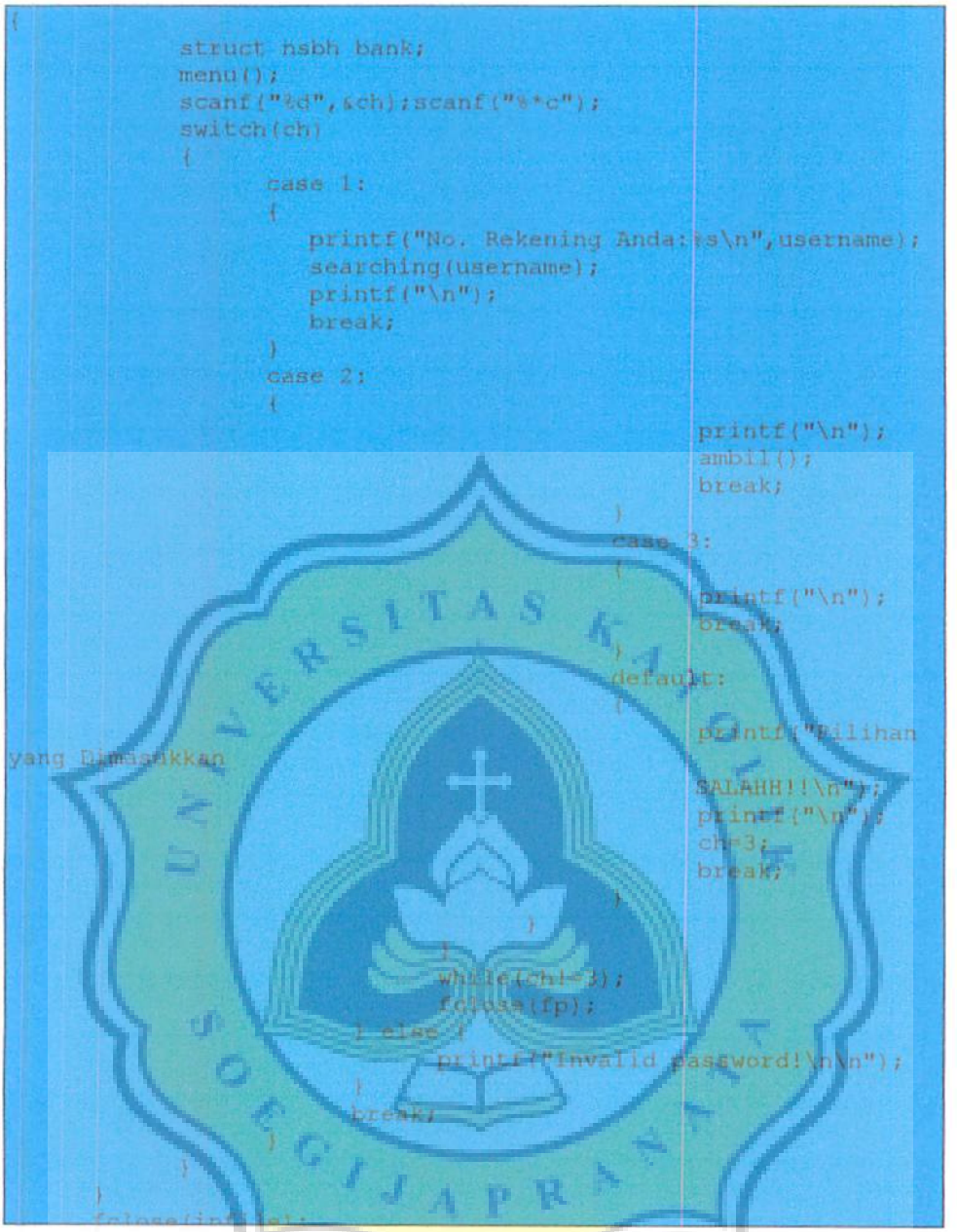
```
int main(int argc, char **argv)
{

    char username[BUFSIZ], *password;
    char buf[BUFSIZ];
    char *user_file, *pass_file;

    FILE *infile;
    printf("Username: ");
    scanf("%s", username);
    password = getpass("Password: ");
    if((infile = fopen("paswd.txt", "r")) == NULL) {
        printf("\nFile error\nAborting...\n");
    } else {
        while(!feof(infile)) {
            buf[0] = '\0';
            fscanf(infile, "%s", buf);
            if(strlen(buf) == 0) continue;
            user_file = buf;
            pass_file = strchr(buf, ';');
            pass_file[0] = '\0';
            pass_file++;
            if(strcmp(user_file, username) == 0){
                if(strcmp(password,pass_file) == 0){
                    printf("Correct password...\n");

                    inithashtable();
                    int ch;
                    char cari[100];
                    do
                    {
```

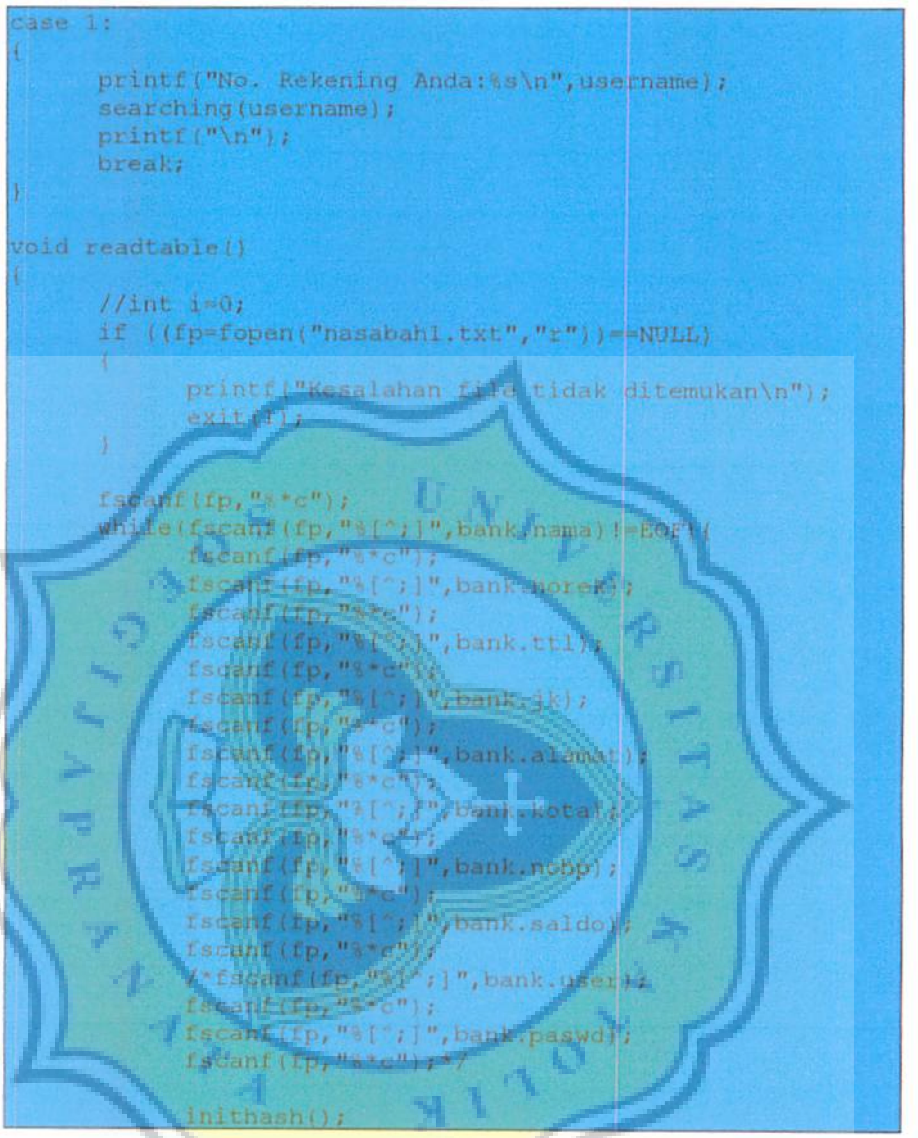
Figure 5.1.1 Coding Username and Password from txt



```
{
        struct nsbh bank;
        menu();
        scanf("%d",&ch);scanf("%*c");
        switch(ch)
        {
              case 1:
              {
                 printf("No. Rekening Anda:%s\n",username);
                 searching(username);
                 printf("\n");
                 break;
              }
              case 2:
              {
                                                    printf("\n");
                                                    ambil();
                                                    break;
                                                 }
                                                 case 3:
                                                 {
                                                    printf("\n");
                                                    break;
                                                 }
                                                 default:
                                                 {
                                                    printf("Pilihan
yang Dimasukkan
                                                    SALAHH!!\n");
                                                    printf("\n");
                                                    ch=3;
                                                    break;
                                                 }
                                              }
                                           }
                                        while(ch!=3);
                                        fclose(fp);
                                     } else {
                                        printf("Invalid password!\n\n");
                                     }
                                     break;
                                  }
                               }
                            }
                            fclose(in
```

Figure 5.1.1 Coding Username and Password

5.1.2 Next step, menu number 1 for displaying user account data taken from search(username) where this function can work with some function readtable() for read from 2 file txt with compare "norek"



```
case 1:
{
      printf("No. Rekening Anda:%s\n",username);
      searching(username);
      printf("\n");
      break;
}

void readtable()
{
      //int i=0;
      if ((fp=fopen("nasabah1.txt","r"))==NULL)
      {
            printf("Kesalahan file tidak ditemukan\n");
            exit(0);
      }

      fscanf(fp,"%*c");
      while(fscanf(fp,"%[^;]",bank.nama)!=EOF){
            fscanf(fp,"%*c");
            fscanf(fp,"%[^;]",bank.norek);
            fscanf(fp,"%*c");
            fscanf(fp,"%[^;]",bank.ttl);
            fscanf(fp,"%*c");
            fscanf(fp,"%[^;]",bank.jk);
            fscanf(fp,"%*c");
            fscanf(fp,"%[^;]",bank.alamat);
            fscanf(fp,"%*c");
            fscanf(fp,"%[^;]",bank.kota);
            fscanf(fp,"%*c");
            fscanf(fp,"%[^;]",bank.nobp);
            fscanf(fp,"%*c");
            fscanf(fp,"%[^;]",bank.saldo);
            fscanf(fp,"%*c");
            /*fscanf(fp,"%[^;]",bank.user);
            fscanf(fp,"%*c");
            fscanf(fp,"%[^;]",bank.paswd);
            fscanf(fp,"%*c");*/

            inithash();
```

Figure 5.1.2 Coding case searching(username) and readtable()

Function intihash() used for take the account number then inserted into a hash table as the result of a hashFunction() with the added rear system linked list.

```
void inithash()
{
      int i;
      i=0;
      i=hashFunction(bank.norek);

      b=hash[i];
      a=malloc(sizeof(bank));
      a->next=NULL;

      strcpy(a->nama,bank.nama);
      strcpy(a->norek,bank.norek);
      strcpy(a->ttl,bank.ttl);
      strcpy(a->jk,bank.jk);
      strcpy(a->alamat,bank.alamat);
      strcpy(a->kota,bank.kota);
      strcpy(a->nohp,bank.nohp);
      strcpy(a->saldo,bank.saldo);
      //strcpy(a->user,bank.user);
      //strcpy(a->paswd,bank.paswd);


      if(b != NULL)
      {
         hash[i]=a;
      }
      else
      {
         while(b->next!=NULL)
         {
               b=b->next;
         }
         b->next=a;
         b=a;
         a->next;
      }

}
```

Figure 5.1.2 Coding inithash()

In the function inithash(), there is hash function that be used to count length of ASCII value from the key value. And then from the total summary of the ASCII value, it will be got the address of index from the modulo 30 (sizeofHash) of ASCII value total summary.

```
int hashFunction(char norek[20])
{
      int nilai,i,val;
      nilai=0;
      for(i=0;i<strlen(norek);i++)
      {
            val=norek[i];
            nilai=nilai+val;
      }
      return nilai%sizeofhash;
}
```

Figure 5.1.2 Coding hashFunction()

And then this is process search(), function a return value (getval) that returns a result modulo between "norek" hash table with the length of the existing.

```
void searching(char key[100])
{
      readtable();
      int getval;
      getval=0;
      getval = hashFunction(key);
      if(hash[getval] != NULL)
      {
            d = hash[getval]; //d utk bantuan looping di hash
ke- getval
            while(d != NULL)
            {
                  if(!(strcmp(key,d->norek)))
                  {
                        printf("Nama: %s\n",d->nama);
                        printf("Saldo Anda: %s\n",d->saldo);
                        printf("\n");
                        amount = strtoul(d->saldo,&p,10);
                        break;
                  }
                  else
                  {
                        d=d->next;
                  }
            }
      }
      else
      {
            printf("MAAF DATA TIDAK DITEMUKAN\n");
            printf("\n");
      }
}
```

Figure 5.1.2 Coding search()

5.1.3 In menu number 2 there are function ambil() used for cash withdrawal there are command amount = strtoul(d->saldo,&p,10);that command for change unsigned long to char.

```
case 2:
{
      printf("\n");
      ambil();
      break;
}


void ambil()
{
      struct nsbh bank;
      unsigned long withdraw;
      printf("Saldo Anda: %lu\n",amount);
      printf("Masukkan jumlah: ");
      scanf("%lu", &withdraw);
      if(amount - withdraw < 100000)
      {
            printf("Saldo tidak mencukupi!\n");
      }
      else
      {
            amount = amount - withdraw;
            printf("Transaksi berhasil\n");
            printf("Saldo Anda: %lu\n",amount);
      }
}
```

Figure 5.1.3 Coding case ambil() and ambil()

5.1.4 In menu number 3 there are function tambah() used for add deposit in user account.

```
void tambah()
{
      unsigned long deposit;
      printf("\n Masukan Jumlah :");


                  scanf("%lu", &deposit);


                        amount = amount + deposit;


                  printf("Saldo Anda : %lu\n", amount);
}
```

Figure 5.1.4 Coding tambah()

5.1.5 And the last menu is the case to exit from the program.

5.2 Testing Program

After explain the coding program, now for running program. First you must to open terminal. Open the folder where there is file .txt. After that running with command gcc -o fixatm fixatm.c then ./fixatm. Input username and password then enter. It will pop up several menu.

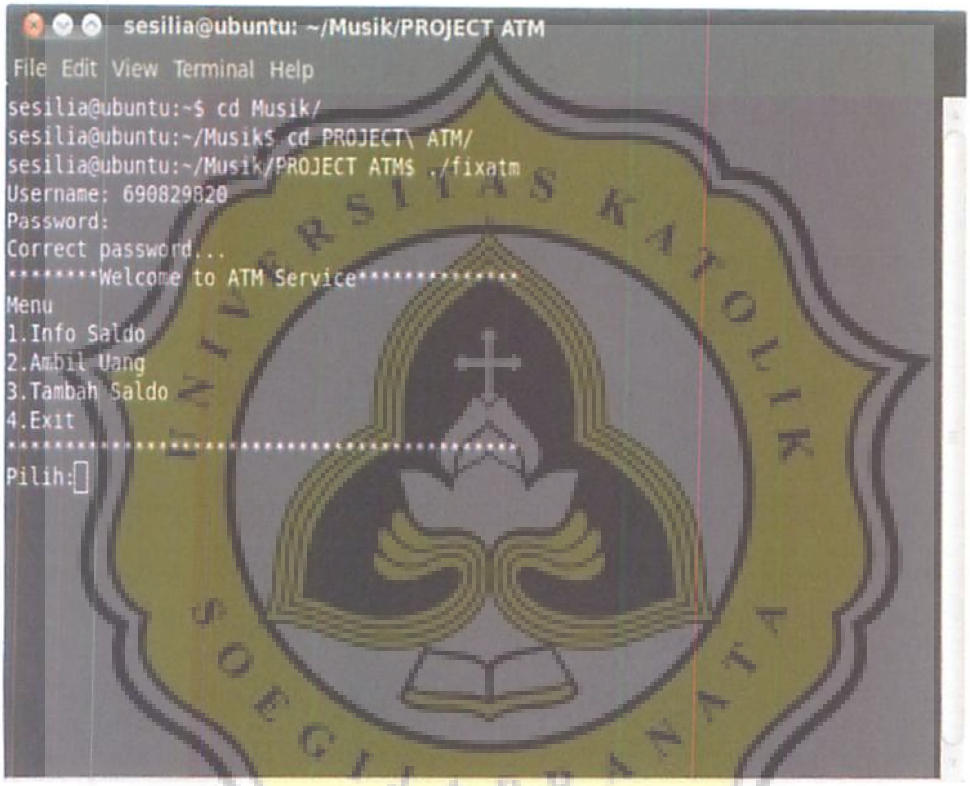


Figure 5.2.1 Login Username and Password

Menu number 1 "Info Saldo", when user choose that it will display account number same as username, name and balancing deposit user.

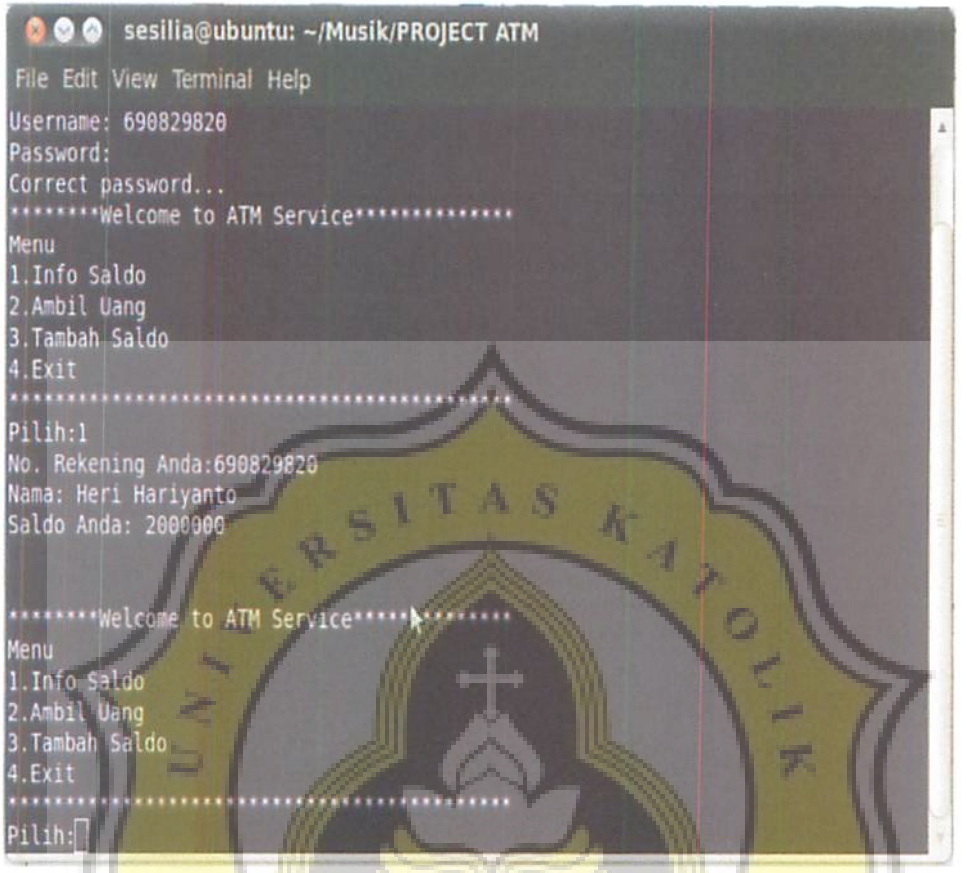


Figure 5.2.2 Process Menu 1 "Info Saldo"

Menu number 2 "Ambil Uang", if user choose this menu, it will display deposit that user has. Then user input amount will be taken. Remaining deposit will appear automatically owned by user.

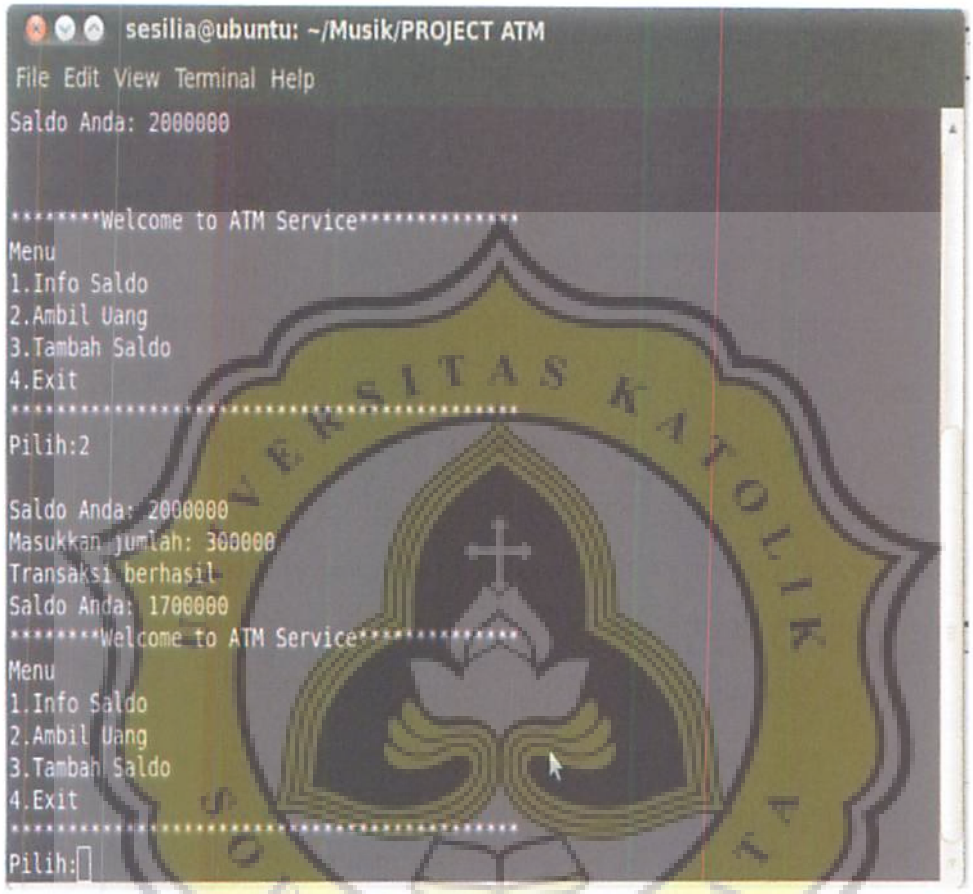


Figure 5.2.3 Process Menu 2 "Ambil Uang"

Menu number 3 "Tambah Saldo" user can input amount for adding deposit and like menu number 2 remaining deposit will appear automatically owned by the user.

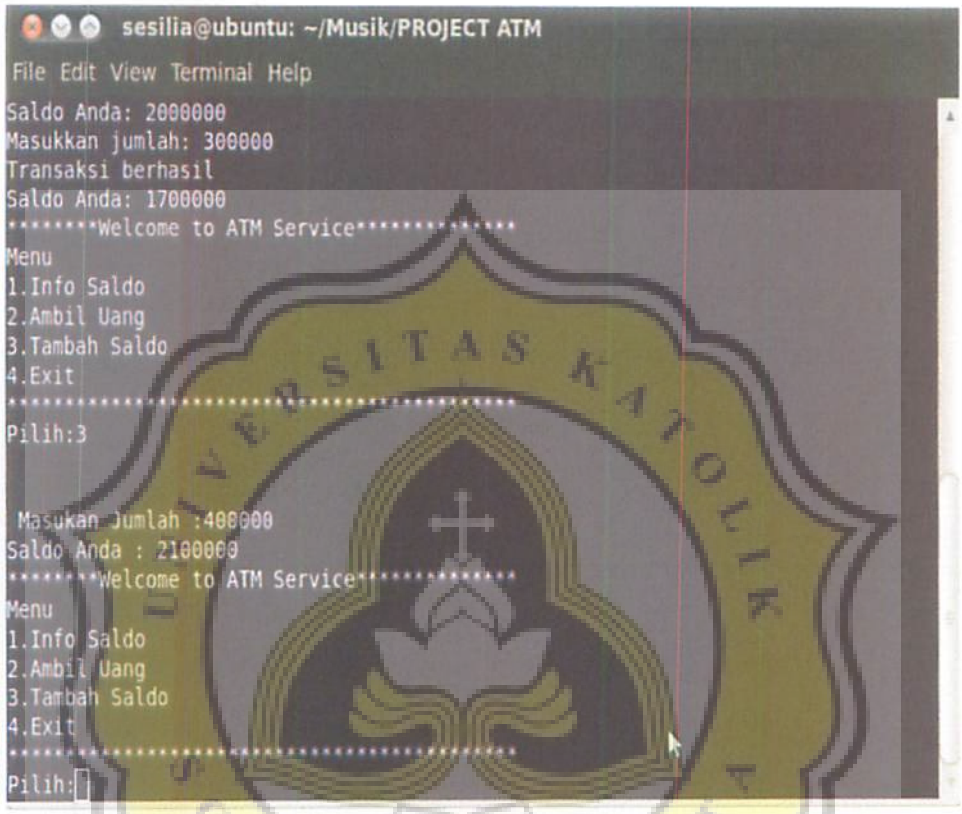


Figure 5.2.4 Process Menu 3 "Tambah Saldo"

Menu number 4 “Exit” user exit from program atm.

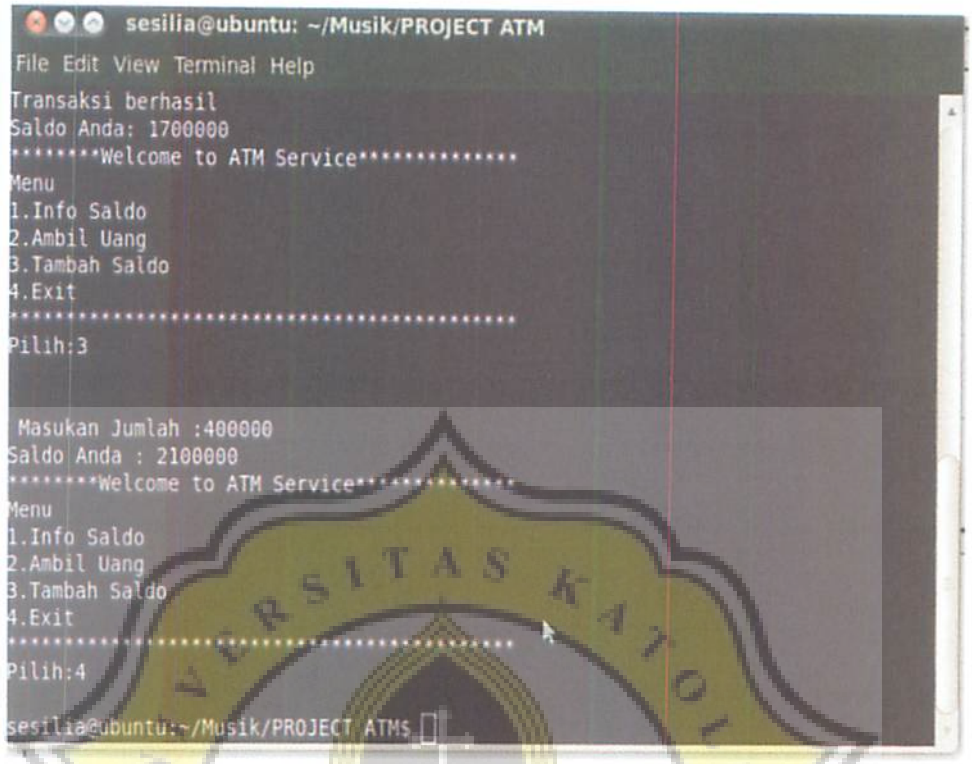


Figure 5.2.5 Process Menu 4 “Exit”