

LHospital

A *Turbo* C++ project

A basic management system for a general hospital

Individual contributions to the project by:

Sankalp Gambhir 9152014

Contents

1	C++ files (.cpp)	2
---	------------------	---

C++ files (.cpp)

1. code/HOSP.CPP

```
1  float hospital::get_balance(){
2      return balance;
3  }
4
5  transaction hospital::deduct_money(float amt, char* reason, Date d, Time t){
6      hospital::balance -= amt;
7
8      ofstream hosp_finances ("transactions.dat", ios::out | ios::binary | ios::app
9          );
10     transaction temp = transaction( (-1)*amt, d, t, reason);
11
12     hosp_finances.write( (char*) (&temp) , sizeof(transaction) );
13
14     hosp_finances.close();
15
16     return temp;
17 }
18
19 transaction hospital::add_money(float amt, char* reason, Date d, Time t){
20     hospital::balance += amt;
21
22     ofstream hosp_finances ("transactions.dat", ios::out | ios::binary | ios::app
23         );
24
25     transaction temp = transaction( (-1)*amt,d, t, reason);
26
27     hosp_finances.write( (char*) (&temp) , sizeof(transaction) );
28
29     hosp_finances.close();
30
31     return temp;
32 }
33 transaction* hospital::get_transaction(){
34     transaction a[10];
35
36     ifstream hosp_finances ("transactions.dat", ios::in | ios::binary);
37
38     hosp_finances.seekg( (-1) * sizeof(transaction) , hosp_finances.end );
39
40     for(int i = 0; i < 10; i++){
41         hosp_finances.read( (char *) &a[i] , sizeof(transaction) );
42         hosp_finances.seekg( hosp_finances.tellg() - ( 2 * sizeof(transaction) )
43             );
44     }
45
46     return a;
47 }
48 patient hospital::get_patient_by_id(long id){
49     patient a;
50
51     str temp;
```

```

52
53     sprintf(temp, "patient/%lu/base.dat", id);
54
55     int i = hospital::read_from(id, (char *) &a, sizeof(patient), temp);
56
57     if(!i){
58         interface::error("File read error!!");
59         getch();
60     }
61
62     return a;
63
64 }
65
66 void hospital::write_patient(patient a){
67     str temp, temp2;
68     sprintf(temp, "patient/%lu/base.dat", a.get_id());
69     sprintf(temp2, "patient/%lu", a.get_id());
70     mkdir("patient");
71     mkdir(temp2);
72     ofstream patient_file ( temp , ios::out | ios::binary );
73
74     if(patient_file){
75         patient_file.write( (char*) &a , sizeof(patient) );
76     }
77     else{
78         interface::error("Patient file access failure!!");
79     }
80     if(patient_file.fail()){
81         interface::error("Patient file write failure!!");
82     }
83     patient_file.close();
84 }
85
86 void hospital::charge_patient(int pat_id, transaction trans){
87     patient temp_pat = hospital::get_patient_by_id(pat_id);
88
89     str temp;
90     sprintf(temp, "patient/%d/trans.dat", temp_pat.get_id());
91     ofstream patient_file ( temp , ios::out | ios::binary | ios::app );
92     patient_file.write( (char*) &trans , sizeof(transaction) );
93     patient_file.close();
94
95     hospital::write_patient(temp_pat);
96 }
97
98 void hospital::discharge_patient(patient temp){
99     temp.discharge();
100     temp.set_discharge_date( system::get_date() );
101     hospital::write_patient(temp);
102 }
103
104 float hospital::calc_bill(int stay){
105     return stay * ::stay_charge;
106 }
107
108 medicine hospital::get_med_by_code(int inp_code){
109     fstream meds ("stock/med.dat", ios::in | ios::binary);
110

```

```

111     medicine temp;
112
113     if(inp_code < 1 || inp_code > 100){
114         temp.code = 0;
115         temp.price = 0;
116         temp.dosage = 0;
117         temp.stock = 0;
118         strcpy(temp.name, "Shell Medicine");
119
120         interface::error("Invalid medicine code!!");
121
122         return temp;
123     }
124
125     for(int i = 0; i<100; i++){
126         mds.read((char*) &temp, sizeof(medicine));
127         if(temp.code == inp_code){
128             break;
129         }
130     }
131
132     return temp;
133 }
134
135 void hospital::write_med(medicine inp_med){
136     fstream med_file ("stock/medicine.dat", ios::in | ios::out | ios::binary);
137     med_file.seekg(0);
138
139     int success = 0;
140
141     while (!success){
142         medicine a;
143         med_file.read( (char*) &a, sizeof(medicine) );
144         if(a.code==inp_med.code){
145             med_file.seekg( med_file.tellg() - sizeof(medicine) );
146             med_file.write( (char*) &a, sizeof(medicine) );
147             success++;
148         }
149     }
150
151 }
152
153 int hospital::get_date_difference(Date dt1, Date dt2)
154 {
155
156     long int n1 = dt1.year*365 + dt1.day;
157
158     for (int i=0; i<dt1.month - 1; i++){
159         n1 += monthDays[i];
160     }
161     n1 += hospital::count_leap_years(dt1);
162
163     long int n2 = dt2.year*365 + dt2.day;
164
165     for (i=0; i<dt2.month - 1; i++){
166         n2 += monthDays[i];
167     }
168     n2 += count_leap_years(dt2);
169

```

```

170     return (n2 - n1);
171 }
172
173 int hospital::count_leap_years(Date d)
174 {
175     int years = d.year;
176
177     if (d.month <= 2){
178         years--;           // checking whether to count the current year
179     }
180
181     return (years / 4) - (years / 100) + (years / 400);
182 }
183
184 int hospital::date_validity(const char * inp_date){
185     return hospital::date_validity(hospital::str_to_date(inp_date));
186 }
187
188 int hospital::date_validity(Date inp_date){
189     if(inp_date.year % 4 == 0 && inp_date.month == 2 &&
190         inp_date.day == 29){
191         return 1;
192     }
193     if (
194         inp_date.month > 12 ||
195         inp_date.day > monthDays[inp_date.month - 1])
196     {
197         return 0;
198     }
199     else{
200         return 1;
201     }
202 }
203
204 Date hospital::str_to_date(const char * inp_date){
205     int counter = 0;
206     int count = 0;
207     int input[3];
208     input[0] = input[1] = input[2] = 0;
209     while(counter < 3){
210         char ch[12];
211         ch[0] = '/';
212         for(int i = 1; i < 7; i++){
213             ch[i] = inp_date[count];
214             count++;
215             if(ch[i] == '/' || ch[i] == '\\\' || ch[i] == 0 || ch[i] == '-'){
216                 if(ch[i] == 0 && count < 11){
217                     interface::error("Invalid date!");
218                     return Date (99, 99, 9999);
219                 }
220                 ch[i] = '/';
221                 int temp = i-1, temp2 = 0;
222                 while(ch[temp] != '/'){
223                     input[counter] += ( pow(10, temp2) * ((int)ch[temp] - (int)'0
224                                     ' ) );
225                     temp--;
226                     temp2++;
227                 }
                counter++;

```

```

228     }
229     }
230 }
231
232     return Date(input[0], input[1], input[2]);
233 }

```

2. code/iface.cpp

```

1 void interface::stock_management(){
2     coord c(ui::scr_width / 3, ui::scr_height / 3);
3     box menu (c, ui::scr_width / 3, ui::scr_height / 2.2);
4
5     int ch = 0;
6
7     menu << "1. Sale"
8         << ui::endl << "2. Purchase"
9         << ui::endl << "3. Stock check"
10        << ui::endl << "4. Go to main menu"
11        << ui::endl << ui::endl << "Choice : ";
12    menu.setdefault(1);
13    menu.settcolor_input(YELLOW);
14    validate_menu::set_menu_limits(1, 4);
15    menu >> validate_menu::input >> ch;
16
17    menu << ui::endl;
18    menu.setexit_button("Submit");
19
20    menu.loop();
21    menu.hide();
22
23    interface::clear_error();
24
25    switch(ch){
26        case 1:
27            {
28
29                medicine temp;
30                temp.code = 0;
31
32                while(temp.code == 0){
33                    coord c(ui::scr_width / 3, ui::scr_height / 3);
34                    box sale_menu (c, ui::scr_width / 3, ui::scr_height / 3);
35                    sale_menu.settcolor_input(YELLOW);
36                    sale_menu << ui::centeralign << "Medicine Sale" << ui::endl;
37                    sale_menu << "Code : ";
38                    sale_menu.setdefault(42);
39                    sale_menu >> temp.code;
40                    sale_menu << ui::endl;
41                    sale_menu.setexit_button("Submit");
42                    sale_menu.loop();
43                    sale_menu.hide();
44
45                    temp = hospital::get_med_by_code(temp.code);
46                }
47
48                int quantity = -2;
49                patient temp_patient;

```



```

50     long pat_id;
51
52     while(quantity < 0 || quantity > 100){
53         coord c(ui::scr_width / 3, ui::scr_height / 3);
54         box sale_menu (c, ui::scr_width / 3, ui::scr_height / 2);
55         sale_menu.settcolor_input(YELLOW);
56         sale_menu << ui::centeralign << "Medicine Sale" << ui::endl;
57         sale_menu << "Name : " << temp.name
58                 << ui::endl << "Price : $" << temp.price
59                 << ui::endl << ui::endl
60                 << "Patient ID : ";
61         sale_menu.setdefault(786);
62         sale_menu >> pat_id;
63         sale_menu << ui::endl << "Quantity : ";
64         sale_menu.setdefault(1);
65         sale_menu >> quantity;
66         sale_menu.setexit.button("Submit");
67         sale_menu.loop();
68         sale_menu.hide();
69
70         temp_patient = hospital::get_patient_by_id(pat_id);
71         if(temp_patient.get_id() == 0){
72             quantity = -1;
73             interface::error("Invalid patient ID!!");
74             continue;
75         }
76         interface::error("Invalid quantity!!");
77     }
78
79     interface::clear_error();
80
81     temp.stock -= quantity;
82
83     for(int i = 0; i < 50; i++){
84         if(temp_patient.get_med(i, 0) == temp.code ||
85            temp_patient.get_med(i, 0) == 0){
86             temp_patient.set_med(i, temp.code, temp_patient.
87                                get_med(i, 1) + quantity);
88         }
89
90         hospital::write_patient(temp_patient);
91         hospital::write_med(temp);
92
93         break;
94     }
95
96     case 2:
97     {
98         medicine temp;
99         temp.code = 0;
100
101         while(temp.code == 0){
102             coord c(ui::scr_width / 3, ui::scr_height / 3);
103             box purchase_menu (c, ui::scr_width / 3, ui::scr_height / 3);
104             purchase_menu.settcolor_input(YELLOW);
105             purchase_menu << ui::centeralign << "Medicine Purchase" << ui::
106                    endl;
107             purchase_menu << "Code : ";

```

```

107         purchase_menu.setdefault(42);
108         purchase_menu >> temp.code;
109         purchase_menu << ui::endl;
110         purchase_menu.setexit_button("Submit");
111         purchase_menu.loop();
112         purchase_menu.hide();
113
114         temp = hospital::get_med_by_code(temp.code);
115     }
116
117     int quantity = -2;
118
119     while(quantity < 0 || quantity > 5000){
120         coord c(ui::scr_width / 3, ui::scr_height / 3);
121         box purchase_menu (c, ui::scr_width / 3, ui::scr_height / 2);
122         purchase_menu.settcolor_input(YELLOW);
123         purchase_menu << ui::centralalign << "Medicine Purchase" << ui::
            endl;
124         purchase_menu << "Name : " << temp.name
125             << ui::endl << "Price : $" << temp.price
126             << ui::endl << ui::endl << "Quantity : ";
127         purchase_menu.setdefault(1);
128         purchase_menu >> quantity;
129         purchase_menu.setexit_button("Submit");
130         purchase_menu.loop();
131         purchase_menu.hide();
132
133         interface::error("Invalid quantity!!");
134     }
135
136     interface::clear_error();
137
138     temp.stock += quantity;
139     hospital::deduct_money(temp.price * quantity, "Medicine purchase",
        system::get_date(), system::get_time());
140     hospital::write_med(temp);
141
142     break;
143 }
144
145 case 3:
146 {
147     medicine temp;
148     temp.code = 0;
149
150     while(temp.code == 0){
151         coord c(ui::scr_width / 3, ui::scr_height / 3);
152         box stock_menu (c, ui::scr_width / 3, ui::scr_height / 3);
153         stock_menu.settcolor_input(YELLOW);
154         stock_menu << ui::centralalign << "Medicine Sale" << ui::endl;
155         stock_menu << "Code : ";
156         stock_menu.setdefault(42);
157         stock_menu >> temp.code;
158         stock_menu << ui::endl;
159         stock_menu.setexit_button("Submit");
160         stock_menu.loop();
161         stock_menu.hide();
162
163         temp = hospital::get_med_by_code(temp.code);

```

```

164         }
165
166         coord c(ui::scr.width / 3, ui::scr.height / 3);
167         box stock_menu (c, ui::scr.width / 3, ui::scr.height / 2);
168         stock_menu.settcolor_input(YELLOW);
169         stock_menu << ui::centralalign << "Medicine Details" << ui::endl;
170         stock_menu << "Name : " << temp.name
171                 << ui::endl << "Price : $" << temp.price
172                 << ui::endl << "Dosage : " << temp.dosage << " ml"
173                 << ui::endl << "Quantity in stock : " << temp.stock
174                 << ui::endl;
175         stock_menu.setexit_button("Okay");
176         stock_menu.loop();
177         stock_menu.hide();
178
179         break;
180     }
181 }
182
183 }
184
185 void interface::error(char* err){
186     window.clear_footer();
187     window.setfooter_tcolor(RED);
188     window << box::setfooter << ui::centralalign
189             << err;
190 }
191
192 void interface::clear_error(){
193     window.clear_footer();
194     window.setfooter_tcolor(GREEN);
195     window << box::setfooter << ui::centralalign
196             << "Everything looks OK";
197 }

```

3. code/iface2.cpp

```

1  #include <fstream.h>
2  #include "base.hpp"
3  #include "iface.hpp"
4  #include "hosp.hpp"
5  #include "emp.hpp"
6
7  void interface::init(){
8      window.hide();
9      window.display();
10     window.settcolor(WHITE);
11     window << ui::centralalign << "LHOSPITAL";
12     window.settcolor(ui::tcolor);
13     window.setfooter_tcolor(GREEN);
14
15     Date current_date = system::get_date();
16     Time current_time = system::get_time();
17
18     str curr_date, curr_time;
19     sprintf(curr_date, "%d/%d/%d", current_date.day, current_date.month,
20             current_date.year);
21     sprintf(curr_time, "%d:%d", current_time.hour, current_time.minute);

```

```

21
22 window << box::setheader << curr_date << box::setheader << ui::rightalign
23     << curr_time << box::setfooter << ui::centeralign
24     << "Everything looks OK";
25 int id;
26 do
27 {
28     id = interface::login.screen();
29     if(id && id.to_emp::convert(id) != OTHERS || id == 1) //so that general
        employees (except administrator) do
30     {
        accidentally login(as they have been assigned // not
31         interface::clear.error(); // username and
        password as "", "")
32         break;
33     }
34 }while(1);
35 if(id == 1) //if user logging in is administrator
36 {
37     int choice = 0;
38
39     while(1){
40         choice = interface::menu();
41
42         switch(choice){
43             case 1:
44                 interface::employee.management();
45                 break;
46             case 2:
47                 interface::stock.management();
48                 break;
49             case 3:
50                 return;
51         }
52     }
53 }
54 else
55 {
56     switch(id.to_emp::convert(id))
57     {
58         case INVALID:
59             interface::error("You have an invalid id generated. Create a new
        account");
60             break;
61         case DOCTOR:
62         case NURSE:
63         case RECEPTIONIST:
64             interface::employee.screen(id);
65             break;
66     }
67 }
68 }
69
70 int interface::login.screen()
71 {
72     const int login.screen.height = 9;
73     coord c(ui::scr.width / 3, ui::scr.height / 3);
74     box login_box (c, ui::scr.width / 3, login.screen.height);
75

```

```

76     str uid, pwd;
77
78     login_box.settcolor_input(YELLOW);
79     login_box << "User ID : ";
80     login_box >> uid;
81     login_box << ui::endl << "Password : ";
82     login_box >> box::setpassword >> pwd;
83     login_box << ui::endl;
84     login_box.setexit_button("Login");
85     login_box.loop();
86     login_box.hide();
87     unsigned long max_id;
88     ifstream fin;
89     fin.open("employee/max_id.dat", ios::binary);
90     if(!fin)
91         max_id = 1;
92     else
93     {
94         fin.read((char *) &max_id, sizeof(unsigned long));
95         if(fin.fail())
96         {
97             interface::error("ERROR WHILE READING FROM FILE!!! ");
98             getch();
99             return 0;
100         }
101     }
102     fin.close();
103     void * x = malloc( sizeof(doctor) );
104     for(unsigned long id = 1; id <= max_id; ++id)
105     {
106         if(x == NULL)
107         {
108             interface::log_this("interface::login_screen() : Not enough memory to
109                 allocate buffer void * temp = malloc( sizeof(doctor) );");
110             interface::error("Out of memory!! Check log");
111             getch();
112             return 0;
113         }
114         if(!hospital::get_employee_by_id(id, x))
115         {
116             char log_msg[300];
117             sprintf(log_msg, "interface::login_screen() : Error in reading file
118                 of id %lu (hospital::get_employee_by_id(id, x) returned 0), could
119                 be due to invalid login details entered", id);
120             interface::log_this(log_msg);
121         }
122         employee * e = (employee *)x;
123         if(!strcmp(e->account.get_username(), uid) && e->account.login(pwd))
124         {
125             interface::clear_error();
126             free(x);
127             return id;
128         }
129     }
130     interface::error("Invalid login details!!");
131     free(x);
132     return 0;
133 }

```

```

132 int interface::menu(){
133     coord c(ui::scr_width / 3, ui::scr_height / 3);
134     box menu (c, ui::scr_width / 3, ui::scr_height / 2.2 + 1);
135
136     int ch;
137     menu << ui::endl << "1. Employee management"
138         << ui::endl << "2. Stock management"
139         << ui::endl << "3. Exit"
140         << ui::endl << ui::endl << "Choice : ";
141     menu.settcolor_input(YELLOW);
142     validate_menu::set_menu_limits(1, 3);
143     menu >> validate_menu::input >> ch;
144
145     menu << ui::endl;
146     menu.setexit_button("Submit");
147
148     menu.loop();
149     menu.hide();
150
151     return ch;
152 }
153
154 void interface::patient_management(){
155     int ch = 0;
156
157     coord c(ui::scr_width / 3, ui::scr_height / 3);
158     box menu (c, ui::scr_width / 3, ui::scr_height / 2.2);
159
160     menu << "1. Patient admission"
161         << ui::endl << "2. Patient discharge"
162         << ui::endl << "3. Edit patient details"
163         << ui::endl << "4. Go to main menu"
164         << ui::endl << ui::endl << "Choice : ";
165     menu.setdefault(1);
166     menu.settcolor_input(YELLOW);
167     validate_menu::set_menu_limits(1,4);
168     menu >> validate_menu::input >> ch;
169
170     menu << ui::endl;
171     menu.setexit_button("Submit");
172
173     menu.loop();
174     menu.hide();
175
176     switch(ch){
177         case 1:
178         {
179             coord c(ui::scr_width / 4, ui::scr_height / 4);
180             box form (c, ui::scr_width / 2, ui::scr_height / 1.5);
181             form.settcolor_input(YELLOW);
182
183             str inp_name, inp_sex_str, inp_dob_str
184                 , inp_phone, inp_guardname, inp_emer_contact
185                 , inp_emer_phone, inp_insur_expiry, inp_admdate_str;
186
187             address inp_adr;
188             disease inp_dis;
189             insurance inp_insur;
190

```

```

191 form << "Enter data for the patient :" << ui::endl
192     << ui::endl << "Name : ";
193 form >> inp_name;
194
195 form << ui::endl << "Sex : ";
196 form >> inp_sex_str;
197 form << ui::endl << "Key - M/F/T = Male/Female/Trans"
198     << ui::endl << "Date of Birth : ";
199
200 form.setdefault("25/12/1991");
201 form >> hospital::date_validity >> inp_dob_str;
202
203
204 form << ui::endl << "Address"
205     << ui::endl << ui::endl
206     << "\tHouse # : ";
207 form.setdefault("221B");
208 form >> inp_adr.house_no;
209
210 form << ui::endl << "\tStreet : ";
211 form.setdefault("Baker Street");
212 form >> inp_adr.street;
213
214 form << ui::endl << "\tDistrict : ";
215 form.setdefault("Idk");
216 form >> inp_adr.district;
217
218 form << ui::endl << "\tState : ";
219 form.setdefault("London(?)");
220 form >> inp_adr.state;
221
222
223 form << ui::endl << ui::endl
224     << "Phone : ";
225 form.setdefault("1234567890");
226 form >> inp_phone;
227
228
229 form << ui::endl << "Disease"
230     << ui::endl << ui::endl
231     << "\tName : ";
232 form.setdefault("Melanoma");
233 form >> inp_dis.name;
234
235 form << ui::endl << "Type : ";
236 form.setdefault(0);
237 form >> inp_dis.type;
238
239 form << ui::endl << "\tType key : " << ui::endl
240     << "\t0 - Brain\t1 - Heart" << ui::endl
241     << "\t2 - Skin\t3 - Lung" << ui::endl
242     << "\t4 - Bone\t5 - Eye" << ui::endl
243     << "\t6 - Throat\t7 - Teeth" << ui::endl
244     << "\t8 - Stomach\t9 - Blood" << ui::endl
245     << "\t10 - General/full body condition"
246     << ui::endl << "\tSymptoms"
247     << ui::endl << "\tSymptom 1 : ";
248
249 form >> inp_dis.symptoms[0];

```

```

250
251     form << ui::endl << "\tSymptom 2 : ";
252     form >> inp_dis.symptoms[1];
253
254     form << ui::endl << "\tSymptom 3 : ";
255     form >> inp_dis.symptoms[2];
256
257     form << ui::endl << "\tSymptom 4 : ";
258     form >> inp_dis.symptoms[3];
259
260
261     form << ui::endl << ui::endl
262         << "Guardian name : ";
263     form.setdefault("Dr. John Watson");
264     form >> inp_guard_name;
265
266     form << ui::endl << "Emergency Contact : ";
267     form.setdefault("Irene Adler");
268     form >> inp_emer_contact;
269
270     form << ui::endl << "Emer. Cont. Phone : ";
271     form.setdefault("1234567890");
272     form >> inp_emer_phone;
273
274
275     form << ui::endl << "Insurance"
276         << ui::endl << ui::endl
277         << "\tProvider : ";
278     form.setdefault("LIC");
279     form >> inp_insur.provider;
280
281     form << ui::endl << "\tAmount ($) : ";
282     form.setdefault(30000);
283     form >> inp_insur.amount;
284
285     form << ui::endl << "\tExpiry";
286     form.setdefault("25/12/2022");
287     form >> hospital::date-validity >> inp_insur.expiry;
288
289
290     form << ui::endl << ui::endl
291         << "Admission Date : ";
292     char dnow[11];
293     form.setdefault("01/01/2018");
294     form >> hospital::date-validity >> inp_admdate_str;
295
296     form << ui::endl << ui::endl;
297     form.setexit_button("Submit");
298
299     form.loop();
300
301     form.hide();
302
303     inp_insur.expiry = hospital::str-to-date(inp_insur.expiry);
304
305     patient temp_pat = patient(inp_name, hospital::str-to-sex(inp_sex_str
306                                     )
                                     , hospital::str-to-date(inp_dob_str),
                                     inp_adr

```



```

307         , inp_phone, inp_dis, inp_guardname
308         , inp_emer_contact, inp_emer_phone
309         , inp_insur, hospital::str_to_date(
            inp_admdate_str));
310
311     hospital::write_patient(temp_pat);
312
313     coord d(ui::scr_width / 3, ui::scr_height / 3);
314     box message (d, ui::scr_width / 3, ui::scr_height / 3);
315
316     message << ui::centralalign << "Patient has been admitted with ID #"
317         << temp_pat.get_id() << ui::endl << ui::endl;
318
319     message.setexit_button("Okay");
320     message.loop();
321     message.hide();
322
323     break;
324 }
325
326 case 2:
327 {
328     patient temp_patient;
329
330     while(1){
331         coord c(ui::scr_width / 3, ui::scr_height / 3);
332         box login_box (c, ui::scr_width / 3, ui::scr_height / 2.5);
333
334         long inp_pat_id;
335
336         login_box << ui::endl << "Patient Discharge"
337             << ui::endl << "Enter patient ID : ";
338         login_box.setdefault(1);
339         login_box >> inp_pat_id;
340
341         login_box << ui::endl;
342         login_box.setexit_button("Submit");
343
344         login_box.loop();
345
346         login_box.hide();
347
348         temp_patient = hospital::get_patient_by_id(inp_pat_id);
349
350         if(temp_patient.get_id() == inp_pat_id){
351             break;
352             interface::clear_error();
353         }
354         else{
355             interface::error("Invalid Patient ID!!");
356             continue;
357         }
358     }
359
360     coord c(ui::scr_width / 3, ui::scr_height / 3);
361     box bill (c, ui::scr_width / 3, ui::scr_height / 2);
362
363     str tt;
364     sprintf(tt, "%d/%d/%d", temp_patient.get_admission_date(DAY),

```

```

365                                     temp_patient.
                                     get_admission_date
                                     (MONTH),
366                                     temp_patient.
                                     get_admission_date
                                     (YEAR));
367
368                                     interface::log_this(
                                     tt);
369
370     int stay_len = abs( hospital::get_date_difference(
371                                     system::get_date(),
372                                     Date(
373                                     temp_patient.
                                     get_admission_date
                                     (DAY),
374                                     temp_patient.
                                     get_admission_date
                                     (MONTH),
375                                     temp_patient.
                                     get_admission_date
                                     (YEAR)
376                                     )
377                                     ) );
378
379     bill << ui::endl << "Bill for " << temp_patient.get_name()
380     << ui::endl << "1. Stay for "
381     << stay_len << " days" << ui::endl;
382
383     float total_bill;
384     bill.settcolor(GREEN);
385     bill << "$" << ( total_bill += hospital::calc_bill(stay_len) );
386
387     for(int i = 0; i < 50; i++){
388         transaction temp_trans = temp_patient.get_transaction(i);
389
390         if( temp_trans.amount == 0 ){
391             break;
392         }
393
394         bill << i+2 << ". " << temp_trans.reason << ui::endl;
395         bill.settcolor(GREEN);
396         bill << "\t$" << temp_trans.amount << ui::endl;
397         bill.settcolor(ui::tcolor);
398
399         total_bill += temp_trans.amount;
400     }
401
402     bill.settcolor(CYAN);
403     bill << ui::endl << "Final bill : $" << total_bill;
404     bill.settcolor(ui::tcolor);
405     bill.setexit_button("Pay Bill");
406     bill.loop();
407     bill.hide();
408
409     hospital::discharge_patient(temp_patient);
410
411     break;
412 }

```

```

413
414     case 3:
415     {
416         int choice = 0;
417
418         patient temp_patient;
419
420         while(1){
421             coord c(ui::scr_width / 3, ui::scr_height / 3);
422             box login_box (c, ui::scr_width / 3, ui::scr_height / 2.5);
423             login_box.settcolor_input(YELLOW);
424
425             long inp_pat_id;
426
427             login_box << ui::endl << "Patient Data Alteration"
428                 << ui::endl << "Enter patient ID : ";
429             login_box.setdefault(1);
430             login_box >> inp_pat_id;
431
432             login_box << ui::endl;
433             login_box.setexit_button("Submit");
434
435             login_box.loop();
436
437             login_box.hide();
438
439             temp_patient = hospital::get_patient_by_id(inp_pat_id);
440
441             if(temp_patient.get_id() == inp_pat_id){
442                 break;
443                 interface::clear_error();
444             }
445             else{
446                 interface::error("Invalid Patient ID!!");
447                 continue;
448             }
449         }
450
451         while(choice < 1 || choice > 5){
452             coord c(ui::scr_width / 3, ui::scr_height / 3);
453             box menu (c, ui::scr_width / 3, ui::scr_height / 1.5);
454
455             menu << "Choose item to edit:"
456                 << ui::endl << "1. Disease/condition"
457                 << ui::endl << "2. Guardian name"
458                 << ui::endl << "3. Emergency contact"
459                 << ui::endl << "4. Emergency contact no."
460                 << ui::endl << "5. Insurance information"
461                 << ui::endl << ui::endl << "Choice : ";
462             menu.setdefault(1);
463             menu.settcolor_input(YELLOW);
464             menu >> choice;
465
466             menu << ui::endl;
467             menu.setexit_button("Submit");
468
469             menu.loop();
470             menu.hide();
471         }

```

```

472     switch(choice){
473         case 1:
474             {
475                 coord c(ui::scr_width / 3, ui::scr_height / 3);
476                 box edit_screen (c, ui::scr_width / 3, ui::scr_height / 2);
477                 edit_screen.setttcolor.input(YELLOW);
478
479                 edit_screen << "Enter disease/condition for " <<
480                     temp_patient.get_name()
481                     << ui::endl << "Disease : ";
482                 disease temp = temp_patient.get_dis();
483                 edit_screen.setdefault(temp.name);
484                 edit_screen >> temp.name;
485                 edit_screen << ui::endl << "Type : ";
486                 edit_screen.setdefault(temp.type);
487                 edit_screen >> temp.type;
488                 edit_screen << ui::endl << "Type key : " << ui::endl
489                     << "0 - Brain\t1 - Heart" << ui::endl
490                     << "2 - Skin\t3 - Lung" << ui::endl
491                     << "4 - Bone\t5 - Eye" << ui::endl
492                     << "6 - Throat\t7 - Teeth" << ui::endl
493                     << "8 - Stomach\t9 - Blood" << ui::endl
494                     << "10 - General/full body condition"
495                     << ui::endl << ui::endl
496                     << "Symptoms" << ui::endl
497                     << "Symptom 1 : ";
498                 edit_screen.setdefault(temp.symptoms[0]);
499                 edit_screen >> temp.symptoms[0];
500                 edit_screen << ui::endl << "Symptom 2 : ";
501                 edit_screen.setdefault(temp.symptoms[1]);
502                 edit_screen >> temp.symptoms[1];
503                 edit_screen << ui::endl << "Symptom 3 : ";
504                 edit_screen.setdefault(temp.symptoms[2]);
505                 edit_screen >> temp.symptoms[2];
506                 edit_screen << ui::endl << "Symptom 4 : ";
507                 edit_screen.setdefault(temp.symptoms[3]);
508                 edit_screen >> temp.symptoms[3];
509
510                 edit_screen << ui::endl << ui::endl;
511                 edit_screen.setexit.button("Submit");
512
513                 edit_screen.loop();
514
515                 edit_screen.hide();
516
517                 temp_patient.set_dis(temp);
518                 hospital::write_patient(temp_patient);
519             }
520         }
521
522         case 2:
523             {
524                 coord c(ui::scr_width / 3, ui::scr_height / 3);
525                 box edit_screen (c, ui::scr_width / 3, ui::scr_height / 2);
526                 edit_screen.setttcolor.input(YELLOW);
527
528                 edit_screen << "Enter name of guardian for " << temp_patient
529                     .get_name()

```

```

529         << ui::endl << "Guardian Name : ";
530     str temp;
531     edit_screen.setdefault(temp_patient.get_guardian_name());
532     edit_screen >> temp;
533
534     edit_screen << ui::endl << ui::endl;
535     edit_screen.setexit.button("Submit");
536
537     edit_screen.loop();
538
539     edit_screen.hide();
540
541     temp_patient.set_guardian_name(temp);
542     hospital::write_patient(temp_patient);
543
544     break;
545 }
546
547 case 3:
548 {
549     coord c(ui::scr_width / 3, ui::scr_height / 3);
550     box edit_screen (c, ui::scr_width / 3, ui::scr_height / 2);
551     edit_screen.settcolor.input(YELLOW);
552
553     edit_screen << "Enter emergency contact no. for " <<
554         temp_patient.get_name()
555         << ui::endl << "Contact no. : ";
556     str temp;
557     edit_screen.setdefault(temp_patient.get_emergency_contact());
558     edit_screen >> temp;
559
560     edit_screen << ui::endl << ui::endl;
561     edit_screen.setexit.button("Submit");
562
563     edit_screen.loop();
564
565     edit_screen.hide();
566
567     temp_patient.set_emergency_contact(temp);
568     hospital::write_patient(temp_patient);
569
570     break;
571 }
572
573 case 4:
574 {
575     coord c(ui::scr_width / 3, ui::scr_height / 3);
576     box edit_screen (c, ui::scr_width / 3, ui::scr_height / 2);
577     edit_screen.settcolor.input(YELLOW);
578
579     edit_screen << "Enter emergency contact no. for " <<
580         temp_patient.get_name()
581         << ui::endl << "Contact no. : ";
582     phone temp;
583     edit_screen.setdefault(temp_patient.get_emer_contact_no());
584     edit_screen >> temp;
585
586     edit_screen << ui::endl << ui::endl;
587     edit_screen.setexit.button("Submit");

```

```

586         edit_screen.loop();
587
588         edit_screen.hide();
589
590         temp_patient.set_emer_contact_no(temp);
591         hospital::write_patient(temp_patient);
592
593         break;
594     }
595
596     case 5:
597     {
598         coord c(ui::scr_width / 3, ui::scr_height / 3);
599         box edit_screen (c, ui::scr_width / 3, ui::scr_height / 2);
600         edit_screen.settcolor_input(YELLOW);
601
602         edit_screen << "Enter insurance information for " <<
603             temp_patient.get_name()
604                 << ui::endl << "Provider : ";
605         insurance temp = temp_patient.get_insur_info();
606         edit_screen.setdefault(temp.provider);
607         edit_screen >> temp.provider;
608         edit_screen << ui::endl << "Amount (in $) : ";
609         edit_screen.setdefault(temp.amount);
610         edit_screen >> temp.amount;
611         edit_screen << ui::endl << "Expiry date (DD/MM/YYYY): ";
612         char temp_date[11];
613         edit_screen >> hospital::date_validity >> temp_date;
614
615         edit_screen << ui::endl << ui::endl;
616         edit_screen.setexit_button("Submit");
617
618         edit_screen.loop();
619
620         edit_screen.hide();
621
622         temp.expiry = hospital::str_to_date(temp_date);
623         temp_patient.set_insur_info(temp);
624         hospital::write_patient(temp_patient);
625
626         break;
627     }
628
629     }
630
631     break;
632 }
633 case 4:
634 {
635     break;
636 }
637 }
638 }

```

4. code/PATIENT.CPP

```

1 #include "patient.hpp"

```

```

2  #include <fstream.h>
3
4  //////////FUNCTION DEFINITIONS FOR CLASS PATIENT//////////
5
6  patient::patient(str inp1, int inp2 , Date inp3, address inp4, phone inp5,
    disease inp6, str inp7, str inp8, phone inp9, insurance inp10, Date inp11) :
    person(inp1, inp2, inp3, inp4, inp5)    //if date_of_admission is the current
    system date, last argument is not needed
7  {
8      fstream pat ("patient/max_id.dat", ios::in | ios::binary | ios::out);
9      long max_id;
10     pat.read( (char*) &max_id, sizeof(long) );
11     max_id++;
12
13     id = max_id;
14
15     pat.seekp(0);
16     pat.write( (char*) &max_id, sizeof(long) );
17     pat.close();
18
19     dis = inp6;
20     strcpy(guardian_name, inp7);
21     strcpy(emergency_contact, inp8);
22     strcpy(emer_contact_no, inp9);
23     insur_info = inp10;
24
25     admission_date = inp11;
26     Date dnow = system::get_date();
27
28     if( admission_date.day != dnow.day ||
29         admission_date.month != dnow.month ||
30         admission_date.year != dnow.year      )
31     {
32         set_dob(inp3, inp11);
33     }
34     for(int i = 0; i < 50; i++){
35         med[i][0] = med[i][1] = 0;
36     }
37
38     bill_amt = 0;    //bill_amt will be set by doctor after treatment
39     discharged = 0;
40 }
41
42 patient::patient()
43 {
44     id = 0;
45 }
46
47 long patient::get_id()
48 {
49     return id;
50 }
51
52 disease patient::get_dis()
53 {
54     return dis;
55 }
56
57 char* patient::get_guardian_name()

```

```

58 {
59     return guardian_name;
60 }
61
62 char* patient::get_emergency_contact()
63 {
64     return emergency_contact;
65 }
66
67 char* patient::get_emer_contact_no()
68 {
69     return emer_contact_no;
70 }
71
72 insurance patient::get_insur_info()
73 {
74     return insur_info;
75 }
76
77 int patient::get_admission_date(int inp)
78 {
79     switch(inp)
80     {
81         case DAY:
82             return admission_date.day;
83         case MONTH:
84             return admission_date.month;
85         case YEAR:
86             return admission_date.year;
87         default:
88             return 0;
89     }
90 }
91
92 int patient::get_discharge_date(int inp)
93 {
94     switch(inp)
95     {
96         case DAY:
97             return discharge_date.day;
98         case MONTH:
99             return discharge_date.month;
100        case YEAR:
101            return discharge_date.year;
102        default:
103            return 0;
104    }
105 }
106
107 unsigned long patient::get_bill_amt()
108 {
109     return bill_amt;
110 }
111
112 int patient::get_med(int a, int b){
113     return med[a][b];
114 }
115
116 transaction patient::get_transaction(int trans_num){

```



```

117     str temp;
118     transaction trans;
119     sprintf(temp, "patient/%d/trans.dat", this->id);
120     ifstream patient_file ( temp , ios::out | ios::binary | ios::app );
121
122     int i = 0;
123     while ( i<=trans.num && patient_file ){
124         patient_file.read( (char*) &trans , sizeof(transaction) );
125         i++;
126     }
127     if( i!= trans.num ){
128         trans = transaction(0);
129     }
130     patient_file.close();
131     return trans;
132 }
133
134 void patient::set_dis(disease a)
135 {
136     dis = a;
137 }
138
139 void patient::set_guardian_name(char *a)
140 {
141     strcpy(guardian_name, a);
142 }
143
144 void patient::set_emergency_contact(char *a)
145 {
146     strcpy(emergency_contact, a);
147 }
148
149 void patient::set_emer_contact_no(char *a)
150 {
151     strcpy(emer_contact_no, a);
152 }
153
154 void patient::set_insur_info(insurance a)
155 {
156     insur_info = a;
157 }
158
159 void patient::set_admission_date(Date a)
160 {
161     admission_date = a;
162     set_dob(dob, admission_date);
163 }
164
165 void patient::set_bill_amt(unsigned long a)
166 {
167     bill_amt = a;
168 }
169
170 void patient::set_med(int a, int b, int c){
171     med[a][0] = b;
172     med[a][1] = c;
173 }
174
175 void patient::set_discharge_date(Date inp){

```

```
176     discharge_date = inp;
177 }
178
179 void patient::discharge() {
180     discharged = 1;
181 }
```