

بنام خدا



دانشگاه صنعتی شریف
دانشکده‌ی مهندسی کامپیوتر

مستند نهایی پروژه درس طراحی پایگاه داده‌ها

مدرس: دکتر مرتضی امینی

پاییز ۱۳۹۹

اعضای گروه:

حمیدرضا کامکاری

مصطفی اوجاقی

کیان باختری

برای سهولت فهم روابط پروژه، طراحی منطقی که در فاز دوم ارائه شد، در این محل قرار داده شده است:

PatientT (PatientID, FirstName, LastName, Age, Gender, Occupation, Reference, Education, HomeAddress, WorkAddress)
PatientPhonesT (PatientID, PhoneNumber)
FileT (PatientID, CreationDate)
PageT (PatientID, PageNo, CreationDate)
MedicalImagePageT (PatientID, PageNo, Content, ImageType, Reason)
AppointmentPageT (PatientID, PageNo, TreatmentSummary, NextAppointmentDate, WholeAmount, PaidAmount, Date, BeginTime)
PersonalInfoPageT (PatientID, PageNo, GeneralMedicalRecords, DentalRecords, SensitiveMedicine, DoesSmoke, Signature)
OccupiedTimeT (Date, BeginTime, Duration, FromDate, WeekDay, BeginTime)
ReferralOccupiedTimeT (Date, BeginTime, Reason, PatientID)
WeeklyScheduleT (FromDate, ToDate)
AvailableTimeT (FromDate, WeekDay, BeginTime, EndTime)

۱- دامنه‌ی صفات و محدودیت‌های جامعیتی:

با توجه به خرد جهان واقع این پروژه، دامنه‌های صفات به شکل زیر در نظر گرفته شده‌اند (قسمت‌هایی از فایل-creat-table.sql ارائه شده است). در این فایل، می‌توان دامنه‌های صفات، کلیدهای اصلی و کلیدهای فرعی هر جدول را مشاهده کرد. تلاش شده است تا حد امکان محدودیت‌ها مطابق با دنیای واقع باشند.

```
1  create domain GENDER as varchar(6) check (
2      VALUE ~ 'MALE'
3      or VALUE ~ 'FEMALE'
4  );
5
6  create domain EDUCATIONAL_DEGREE as varchar(50) check (
7      VALUE ~ 'High-School-Diploma' or
8      VALUE ~ 'Associate-Degree' or
9      VALUE ~ 'Bachelors-Degree' or
10     VALUE ~ 'Masters-Degree' or
11     VALUE ~ 'Doctoral-Degree' or
12     VALUE ~ 'Other'
13 );
14
15 create table if not exists PatientT
16 (
17     Patient_ID  integer primary key,
18     first_name  varchar(50) not null,
19     last_name   varchar(50) not null,
20     age         integer    not null,
21     gender      GENDER     not null,
22     occupation  varchar(50),
23     reference   varchar(50),
24     education   EDUCATIONAL_DEGREE,
25     homeAddress varchar(100),
26     workAddress varchar(100)
27 );
```

```

30  create domain PHONE as char(11) check (
31      VALUE ~ '0\d{10}'
32  );
33
34  create table if not exists PatientPhonesT
35  (
36      patient_id    integer,
37      phone_number PHONE,
38      primary key (patient_id, phone_number),
39      constraint fk_patient
40          foreign key (patient_id)
41              references patientt (patient_id)
42              on delete cascade
43              on update cascade
44  );
45
46  create table FileT
47  (
48      patient_id    integer primary key,
49      creation_date date,
50      constraint fk_patient
51          foreign key (patient_id)
52              references patientt (patient_id)
53              on delete cascade
54              on update cascade
55  );
56
57  create table pageT
58  (
59      patient_id    integer,
60      page_no       integer,
61      creation_date date,
62      primary key (patient_id, page_no),
63      constraint fk_patient
64          foreign key (patient_id)
65              references patientt (patient_id)
66              on delete cascade
67              on update cascade
68  );
69
70
71  create domain MEDICAL_IMAGE_TYPE as varchar(50)
72      check (
73          VALUE ~ 'CT-Scan' or
74          VALUE ~ 'OPG' or
75          VALUE ~ 'Radiographic-Image'
76      );

```

```

78  create table MedicalImagePageT
79  (
80      patient_id      integer,
81      page_no         integer,
82      content_address varchar(100)      not null,
83      image_type      MEDICAL_IMAGE_TYPE not null,
84      reason          varchar(500),
85      primary key (patient_id, page_no),
86      constraint fk_paget
87          foreign key (patient_id, page_no)
88              references paget (patient_id, page_no)
89              on delete cascade
90              on update cascade
91  );
92

93  create domain WEEK_DAY as char(3) check (
94      VALUE ~ 'SAT' or
95      VALUE ~ 'SUN' or
96      VALUE ~ 'MON' or
97      VALUE ~ 'TUE' or
98      VALUE ~ 'WED' or
99      VALUE ~ 'THU' or
100     VALUE ~ 'FRI'
101 );
102

103 create table WeeklyScheduleT
104 (
105     from_date date primary key,
106     to_date   date not null unique
107 );
108

109 create table AvailableTimeSlotsT
110 (
111     weekly_schedule_from_date_ref date,
112     day_of_week                  WEEK_DAY not null,
113     begin_time                   time    not null,
114     duration                      time    not null,
115     primary key (weekly_schedule_from_date_ref, day_of_week, begin_time),
116     constraint fk_weekly_schedule
117         foreign key (weekly_schedule_from_date_ref)
118             references WeeklyScheduleT (from_date)
119             on delete cascade
120             on update cascade
121 );
122

123 create table OccupiedTimeSlotsT
124 (
125     date                  date,
126     begin_time            time,
127     duration              time    not null,
128     available_time_slots_ref_from_date date    not null,
129     available_time_slots_ref_week_day  WEEK_DAY not null,
130     available_time_slots_ref_begin_time time   not null,
131     primary key (date, begin_time),
132     constraint fk_OccupiedTimeSlotsT
133         foreign key (available_time_slots_ref_from_date, available_time_slots_ref_week_day,
134                         available_time_slots_ref_begin_time)
135         references AvailableTimeSlotsT (weekly_schedule_from_date_ref, day_of_week, begin_time)
136         on delete cascade
137         on update cascade
138 );

```

```

140  create table AppointmentPageT
141  (
142      patient_id          integer,
143      page_no              integer,
144      treatment_summary    varchar(500),
145      next_appointment_date date,
146      whole_payment_amount integer,
147      paid_payment_amount  integer,
148      occupied_time_slot_date_ref date,
149      occupied_time_slot_begin_time_ref time,
150      primary key (patient_id, page_no),
151
152      constraint fk_occupiedtimeslotst
153          foreign key (occupied_time_slot_date_ref, occupied_time_slot_begin_time_ref)
154              references OccupiedTimeSlotsT (date, begin_time)
155              on delete cascade
156              on update cascade,
157
158      constraint fk_paget
159          foreign key (patient_id, page_no)
160              references paget (patient_id, page_no)
161              on delete cascade
162              on update cascade
163  );
164
165  create table PersonalInfoPageT
166  (
167      patient_id          integer,
168      page_no              integer,
169      general_medical_records varchar(500),
170      dental_records       varchar(500),
171      sensitive_medicine   varchar(500),
172      does_smoke           boolean,
173      signature_image_address varchar(100),
174      primary key (patient_id, page_no),
175
176      constraint fk_PageT
177          foreign key (patient_id, page_no)
178              references paget (patient_id, page_no)
179              on delete cascade
180              on update cascade
181  );
182
183  create table ReferralOccupiedTimeSlotsT
184  (
185      date      date,
186      begin_time time,
187      reason    varchar(500),
188      patient_id integer not null,
189      primary key (date, begin_time),
190      constraint fk_patient
191          foreign key (patient_id)
192              references PatientT (patient_id)
193              on delete cascade
194              on update cascade,
195      constraint fk_OccupiedTimeSlotsT
196          foreign key (date, begin_time)
197              references OccupiedTimeSlotsT (date, begin_time)
198              on delete cascade
199              on update cascade
200  );

```

۲- رهاناهای مربوط به صفحه مراجعه:

در این پروژه برای جلوگیری از نقض محدودیت‌های موجود، رهاناهای زیر مورد استفاده قرار گرفته است.

• رهاناهای مربوط به صفحه مراجعه:

```
9  create function appointment_page_trigger_function() returns trigger
10    language plpgsql as
11    $$
12    begin
13      -- check if the paid payment <= whole payment if not revert changes
14      if new.whole_payment_amount < new.paid_payment_amount then
15        raise exception 'paid amount is larger than whole amount in query, %', now();
16      end if;
17      -- check if next appointment date is after the current appointment date if not revert changes
18      if new.next_appointment_date < new.occupied_time_slot_date_ref then
19        raise exception 'next appointment date is not after the current appointment date, %', now();
20      end if;
21      return new;
22    end;
23    $$;
24
25    -- define insert trigger:
26    create trigger appointment_page_trigger
27      before insert or update
28      on AppointmentPageT
29      for each row
30      execute procedure appointment_page_trigger_function();
```

• رهاناهای مربوط به صفحه OccupiedTime:

```
36  create function occupied_time_slots_trigger_function()
37    returns trigger as
38    $$
39    begin
40      -- check if the date matches weekly schedule dates
41      if new.date < new.available_time_slots_ref_from_date or
42        new.date > (select to_date from WeeklyScheduleT where from_date = new.available_time_slots_ref_from_date)
43      then
44        raise exception 'Occupied time date does not match the weekly schedule%', now();
45      end if;
46      -- check if the date matches the available time in schedules week day
47      declare
48        occupied_time_day_of_week_as_int    integer;
49        occupied_time_day_of_week_as_string char(3);
50      begin
51        select extract(dow from new.date) into occupied_time_day_of_week_as_int;
52        case occupied_time_day_of_week_as_int
53        when 0 then
54          occupied_time_day_of_week_as_string := 'SUN';
55        when 1 then
56          occupied_time_day_of_week_as_string := 'MON';
57        when 2 then
58          occupied_time_day_of_week_as_string := 'TUE';
59        when 3 then
60          occupied_time_day_of_week_as_string := 'WED';
61        when 4 then
62          occupied_time_day_of_week_as_string := 'THU';
63        when 5 then
64          occupied_time_day_of_week_as_string := 'FRI';
65        when 6 then
66          occupied_time_day_of_week_as_string := 'SAT';
67        end case;
68        if occupied_time_day_of_week_as_string != new.available_time_slots_ref_week_day then
69          raise exception 'Occupied time date does not match the day of week field %', now();
70        end if;
71      end;
72      -- check if the date matches the available time in schedules time interval in the day
```

```

73 declare
74     available_time_record record;
75 begin
76     select duration, begin_time
77     from AvailableTimeSlotsT
78     where new.available_time_slots_ref_from_date = weekly_schedule_from_date_ref
79     and new.available_time_slots_ref_week_day = day_of_week
80     and new.available_time_slots_ref_begin_time = begin_time into available_time_record;
81
82     if new.available_time_slots_ref_begin_time > new.begin_time or
83         available_time_record.duration::interval + available_time_record.begin_time < new.begin_time + new.duration::interval then
84         raise exception 'Occupied time in day does not match with the available time %', now();
85     end if;
86 end;
87
88 -----
89 if exists(select *
90           from occupiedtimeslotst
91           where date = new.date
92             and (begin_time < new.begin_time and new.begin_time < begin_time + duration::interval
93                   or begin_time < new.begin_time+new.duration::interval and new.begin_time+new.duration::interval < begin_time + duration::interval)) then
94     raise exception 'Occupied Times should not overlap %', now();
95 end if;
96
97     return new;
98 end;
99 $$*
100 language plpgsql;
101
102 create trigger occupied_time_slot_trigger
103     before insert or update
104     on OccupiedTimeSlotsT
105     for each row
106 execute procedure occupied_time_slots_trigger_function();

```

رهانهای مربوط به صفحه مشخصات شخصی و پرونده:

```

109 create function personal_info_page_trigger_function()
110     returns trigger as
111 $$*
112 begin
113     if new.page_no != 1 then
114         raise exception 'Personal info page number should be 1 %', now();
115     end if;
116     return new;
117 end
118 $$ language plpgsql;
119
120 create trigger personal_info_page_trigger
121     before update or insert
122     on personalinfopaget
123     for each row
124 execute procedure personal_info_page_trigger_function();
125
126 create function page_no_trigger_function()
127     returns trigger as
128 $$*
129 begin
130     if new.page_no != 1 and not exists(select * from paget where paget.patient_id = new.patient_id and paget.page_no = new.page_no - 1) then
131         raise exception 'Page Numbers should be consecutive %', now();
132     end if;
133     return new;
134 end
135 $$ language plpgsql;
136
137 create trigger page_no_trigger
138     before update or insert
139     on paget
140     for each row
141 execute procedure page_no_trigger_function();
142

```

• رهاناهای مربوط به AvailableTime و WeeklySchedule :ها

```
144 create function weekly_schedule_trigger_function()
145     returns trigger as
146     $$
147 begin
148     if exists (select * from weeklyschedule
149                 where (from_date < new.from_date and new.from_date <= to_date)
150                     or (from_date <= new.to_date and new.to_date <= to_date)) then
151         raise exception 'Weekly schedules should not overlap %', now();
152     end if;
153
154     if new.to_date < new.from_date then
155         raise exception 'End date can not be before start date.';
156     end if;
157     return new;
158 end
159 $$ language plpgsql;
160
161 create trigger weekly_schedule_trigger
162     before update or insert
163     on weeklyschedule
164     for each row
165 execute procedure weekly_schedule_trigger_function();
166
167
168 create function available_time_trigger_function()
169     returns trigger as
170     $$
171 begin
172     if exists(select *
173                 from availabletimeslotst
174                 where weekly_schedule_from_date_ref = new.weekly_schedule_from_date_ref
175                     and day_of_week = new.day_of_week
176                     and (begin_time < new.begin_time and new.begin_time < begin_time + duration::interval
177                         or begin_time < new.begin_time+new.duration::interval and new.begin_time+new.duration::interval < begin_time + duration::interval)) then
178         raise exception 'Available Times should not overlap %', now();
179     end if;
180     return new;
181 end
182 $$ language plpgsql;
183
184 create trigger available_time_trigger
185     before update or insert
186     on availabletimeslotst
187     for each row
188 execute procedure available_time_trigger_function();
```

رهانهای مربوط به شماره صفحات پرونده و یکتایی آن‌ها:

```
191  create function delete_page_trigger_function()
192    returns trigger as
193  $$
194  begin
195    if old.page_no = 1 or exists (select * from paget where patient_id = new.patient_id and page_no = new.page_no + 1) then
196      raise exception 'Personal info page number should be 1 %', now();
197    end if;
198    return new;
199  end
200  $$ language plpgsql;
201
202  create trigger delete_page_trigger
203    before delete
204    on paget
205    for each row
206  execute procedure delete_page_trigger_function();
207
208
209  create function page_uniqueness_trigger_function()
210    returns trigger as
211  $$
212  begin
213    if new.page_no = 1 then
214      raise exception 'Page number 1 is reserved for Personal info page %', now();
215    end if;
216    if exists(select * from appointmentpaget where patient_id = new.patient_id and page_no = new.page_no)
217      and exists(select * from medicalimagepaget where patient_id = new.patient_id and page_no = new.page_no)
218    then
219      raise exception 'There is another page with this number %', now();
220    end if;
221    return new;
222  end
223  $$ language plpgsql;
224
225  create trigger appointment_page_uniqueness_trigger
226    before insert
227    on appointmentpaget
228    for each row
229  execute procedure page_uniqueness_trigger_function();
230
231  create trigger medical_image_page_uniqueness_trigger
232    before insert
233    on medicalimagepaget
234    for each row
235  execute procedure page_uniqueness_trigger_function();
236
237  create function update_page_no_trigger_function()
238    returns trigger as
239  $$
240  begin
241    raise exception 'Page number can not be updated %', now();
242    return new;
243  end
244  $$ language plpgsql;
245
246  create trigger appointment_page_update_trigger
247    before update
248      of page_no
249    on appointmentpaget
250    for each row
251  execute procedure update_page_no_trigger_function();
252
253  create trigger medical_image_page_update_trigger
254    before update
255      of page_no
256    on medicalimagepaget
257    for each row
258  execute procedure update_page_no_trigger_function();
```

۳- پرس‌وجوها:

با توجه به نیازمندی‌های پژوهه و پرس‌وجوهای استخراج شده در فاز صفرم، پرس‌وجوهای زیر طراحی شده و توسط application ای که با استفاده از زبان جاوا نوشته شده است، به پایگاه داده ارسال می‌شوند. در بخش بعدی در مورد application توضیحات بیشتری ارائه خواهیم داد.

پرس‌وجوها از پایگاه داده:

- ایجاد یک پرونده جدید برای یک بیمار جدید (شامل ایجاد یک «بیمار» جدید و «پرونده» برای او)
- اضافه کردن یک «صفحه مراجعه» به یک پرونده
- اضافه کردن یک مرجعه جدید (برای نوبت‌دهی توسط منشی انجام می‌شود)
- اصلاح یک صفحه از یک پرونده
- لغو یک مرجعه/بیمار/پرونده از پایگاه داده (لغو بیمار یا پرونده به علت حساسیت بالا، نیازمند double checking خواهد بود، به این منظور که صرفاً با یک خطای منشی، پرونده هیچ بیماری حذف نشود)
- فراخوانی لیست بیماران
- فراخوانی لیست مراجعات قبلی و آینده (هم تمامی مراجعات، و هم مراجعات یک بیمار خاص)
- فراخوانی پرونده بیماران
- فراخوانی زمان پیشنهادی بعدی برای مرجعه یک بیمار
- فراخوانی صورت مالی (پرداخت‌ها) در هر بازه‌ی زمانی قبلی
- درخواست لغو کردن همه‌ی مراجعات آینده در یک بازه‌ی زمانی خاص و نام و شماره تلفن بیماران مربوطه برای خبر دادن به آن‌ها
- فراخوانی همه‌ی زمان‌های تخصیص داده نشده به مراجعات برای هر بازه‌ی زمانی انتخابی در آینده (جهت نوبت‌دهی به سایر مراجعان)
- فراخوانی همه‌ی بیمارانی که در یک بازه زمانی خاص، یک فعالیت خاص برایشان انجام شده است. برای مثال همه‌ی بیمارانی که در ماه اخیر عصب‌کشی داشته‌اند.
- فراخوانی همه‌ی مراجعات Follow up در یک بازه‌ی زمانی خاص در آینده
- فراخوانی همه‌ی بیمارانی که در یک بازه‌ی زمانی خاص بیشتر از n بار مراجعه داشته‌اند
- فراخوانی بیمارانی که معرفشان یک بیمار خاص بوده است
- فراخوانی مراجعاتی که در یک بازه‌ی زمانی خاص به یک علت خاص بوده‌اند
- فراخوانی همه‌ی مراجعات Follow up در یک بازه‌ی زمانی خاص در آینده که قرار است فعالیتی خاص برایشان صورت گیرد.
- فراخوانی بیماران در یک بازه‌ی سنی خاص

۴- نرمال سازی

از ابتدا تلاش شد تا روابط نرمال باشند. در نهایت پس از یک بررسی کلی و تغییراتی اندک، روابط این پروژه به سطح BCNF رسیدند.

۵- دیدها:

طبق نیازمندی هایی که در فاز صفرم پروژه تشخیص داده شدند، دیدهایی برای کاربر سایت طراحی شده است که دید کاربر سایت را محدود می کنند و در مواردی که لازم باشد، از اصلاح شدن جداول توسط کاربر جلوگیری می کند (با استفاده از رهانا

پیاده سازی شده است). پرس و جوهایی که کاربر سایت مورد استفاده قرار می دهد، به صورت Stored procedure نوشته و ذخیره شده است:

```
1  create view WeeklyScheduleV as select * from weeklyschedule;
2  create view AvailableTimeSlotsV as select * from availabletimeslotst;
3  create view OccupiedTimeSlotsV as select * from occupiedtimeslotst;
4  create view ReferralOccupiedTimeSlotsV as select * from referraloccupiedtimeslotst;
5
6  create function do_not_change()
7      returns trigger
8  as
9  $$
10 begin
11     raise exception 'Cannot modify table.
12 Contact the system administrator if you want to make this change.';
13 end;
14 $$
15 language plpgsql;
16
17 create trigger weekly_schedule_view_trigger
18     before insert or update or delete
19     on WeeklyScheduleV
20 execute procedure do_not_change();
21
22 create trigger available_time_slots_view_trigger
23     before insert or update or delete
24     on AvailableTimeSlotsV
25 execute procedure do_not_change();
26
27 create trigger occupied_time_slots_view_trigger
28     before delete
29     on OccupiedTimeSlotsV
30 execute procedure do_not_change();
31
32 create trigger referral_view_trigger
33     before update or delete
34     on ReferralOccupiedTimeSlotsV
35 execute procedure do_not_change();
36
37
38 create function available_time_in_date(d date)
39     returns table(begin_time time, duration time) as
40 $$
41 declare
42     occupied_time_day_of_week_as_int    integer;
43     occupied_time_day_of_week_as_string char(3);
44 begin
45     select extract(dow from d) into occupied_time_day_of_week_as_int;
46     case occupied_time_day_of_week_as_int
47         when 0 then
48             occupied_time_day_of_week_as_string := 'SUN';
49         when 1 then
50             occupied_time_day_of_week_as_string := 'MON';
51         when 2 then
52             occupied_time_day_of_week_as_string := 'TUE';
53         when 3 then
54             occupied_time_day_of_week_as_string := 'WED';
55         when 4 then
```

```

56     occupied_time_day_of_week_as_string := 'THU';
57     when 5 then
58         occupied_time_day_of_week_as_string := 'FRI';
59     when 6 then
60         occupied_time_day_of_week_as_string := 'SAT';
61     end case;
62 end;
63 return query
64 select T1.begin_time, T1.duration from WeeklyScheduleV natural join AvailableTimeSlotsV as T1
65 where from_date <= d
66 and d <= to_date
67 and occupied_time_day_of_week_as_string = day_of_week;
68 end;
69 $$ language plpgsql;
70
71 create function occupied_time_in_date(d date)
72 returns table(begin_time time, duration time) as
73 $$ begin
74 begin
75     return query
76     select OccupiedTimeSlotsV.begin_time, OccupiedTimeSlotsV.duration
77     from OccupiedTimeSlotsV
78     where OccupiedTimeSlotsV.date <= d;
79 end;
80 $$ language plpgsql;
81
82
83
84 create function reserve_time(d date, btime time, dur time, rsn varchar(500), pid integer)
85 returns integer as
86 $$ declare
87 occupied_time_day_of_week_as_int    integer;
88 occupied_time_day_of_week_as_string char(3);
89 available record;
90 begin
91     select extract(dow from d) into occupied_time_day_of_week_as_int;
92     case occupied_time_day_of_week_as_int
93     when 0 then
94         occupied_time_day_of_week_as_string := 'SUN';
95     when 1 then
96         occupied_time_day_of_week_as_string := 'MON';
97     when 2 then
98         occupied_time_day_of_week_as_string := 'TUE';
99     when 3 then
100        occupied_time_day_of_week_as_string := 'WED';
101    when 4 then
102        occupied_time_day_of_week_as_string := 'THU';
103    when 5 then
104        occupied_time_day_of_week_as_string := 'FRI';
105    when 6 then
106        occupied_time_day_of_week_as_string := 'SAT';
107    end case;
108 select T1.weekly_schedule_from_date_ref, T1.day_of_week, T1.begin_time from WeeklyScheduleV natural join AvailableTimeSlotsV as T1
109 where from_date <= d
110 and d <= to_date
111 and occupied_time_day_of_week_as_string = day_of_week
112
113     and T1.begin_time <= btime and btime < T1.begin_time + T1.duration::interval
114     and btime + dur::interval < T1.begin_time + T1.duration::interval into available;
115 insert into OccupiedTimeSlotsV values (d, btime, dur,
116                                         available.weekly_schedule_from_date_ref,
117                                         available.day_of_week,
118                                         available.begin_time);
119 insert into ReferralOccupiedTimeSlotsV values (d, btime, rsn, pid);
120 return 1;
121 end;
122 $$ language plpgsql;

```

۶- پشتیبان‌گیری خودکار:

برای این منظور ذخیره‌سازی `Wal` فعال شده که بازیابی نقطه‌ای را ممکن می‌کند.

Application -۷

برای برقراری ارتباط با پایگاه داده، یک واسط گرافیکی با استفاده از زبان جاوا طراحی شده است که حاوی هر سه بخش `View`, `Model` و `Controller` می‌باشد. پرس‌وجوهای پروژه از طریق واسط گرافیکی به بخش `Controller` منتقل شده و از پایگاه داده درخواست می‌شود. نتیجه بازیابی شده و به بخش گرافیکی منتقل می‌شود تا مورد استفاده قرار بگیرد. در زیر، تصاویری از GUI پروژه ارائه شده است.

First Name	Last Name	Patient ID
hamid kamkari	Patient id: 1	
Kian Bakhtari	Patient id: 2	
Mostafa Ojaghi	Patient id: 3	
GhamarOlmolook vaziri	Patient id: 4	
Hassan Kilid	Patient id: 5	
Morteza Amini	Patient id: 6	
Abbas Abbaszadeh	Patient id: 7	
Mohammad Mohammadi	Patient id: 8	
Sdegh Kesha	Patient id: 9	
Ostad Raefipoor	Patient id: 10	
kik BB	Patient id: 15	
kik BB	Patient id: 16	
kik BB	Patient id: 17	
kik BB	Patient id: 18	
fn ln	Patient id: 20	

Files

First Name	Last Name	Patient ID	<input checked="" type="checkbox"/> In Debt / Not In Debt	Search
GhamarOlmolook vaziri		Debt value: 100	Patient id: 4	
Add New File				

Files

First Name	Last Name	Patient ID	<input type="checkbox"/> In Debt / Not In Debt	Search
kik			<input type="checkbox"/>	
kik BB		Patient id: 15		
kik BB		Patient id: 16		
kik BB		Patient id: 17		
kik BB		Patient id: 18		
Add New File				

Files

Add New File:

Patient First Name:

Patient Last Name:

Age:

Gender:

Occupation:

Reference:

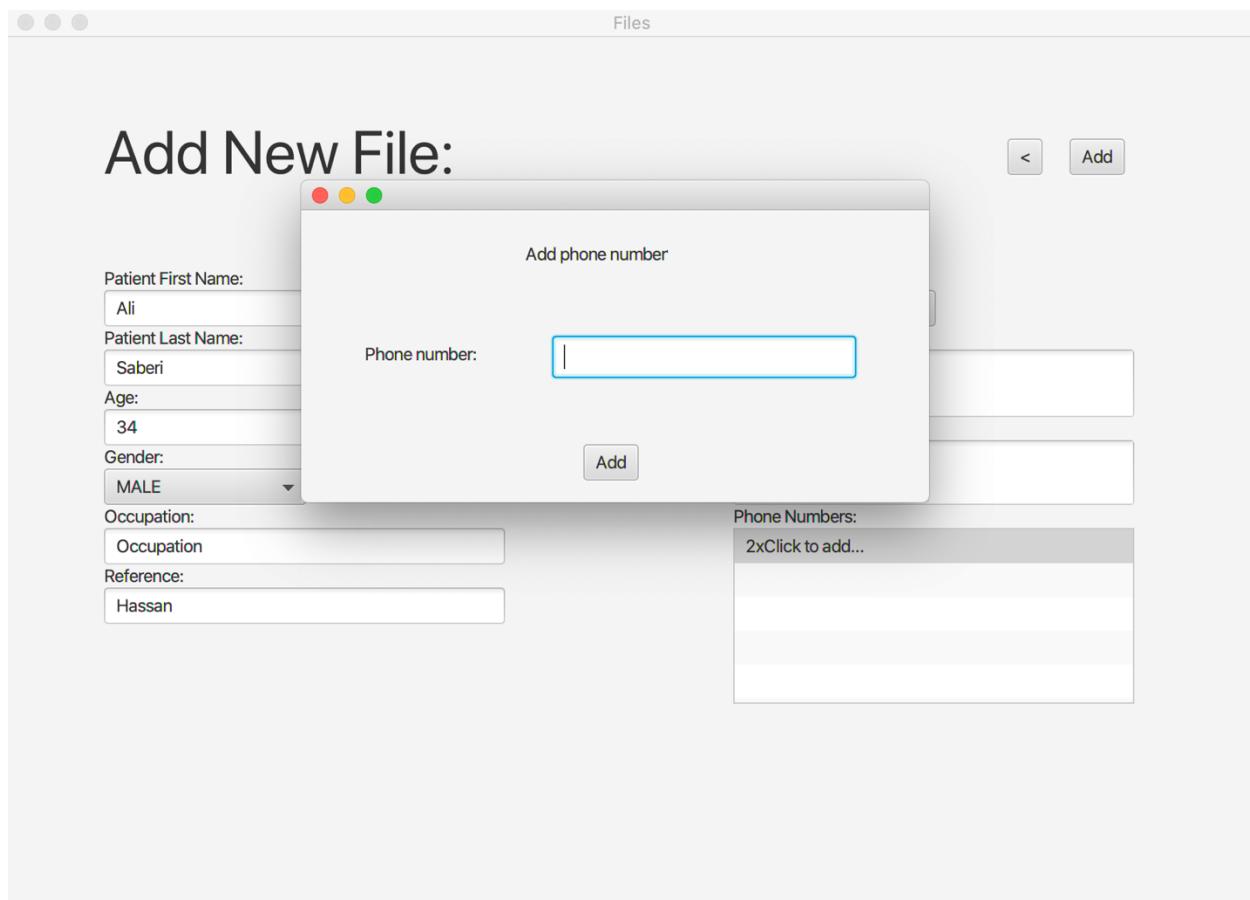
Education

Home Address:

Work Address:

Phone Numbers:
2xClick to add...

< Add



Files

ID: 8
Pages

Personal Info 1

Mohammad Mohammdi

Patient ID: 8

upload photo

First Name: Mohammad
Last Name: Mohammdi
Age: 20
Gender: MALE
Occupation: student
Reference: someReff
Education: High-School-Diploma

Home Address: mamadAddr
Work Address: mamadAddr

General Medical Records: p8 general medical records
Dental Records: p8 dental records
Sensitive Medicine: p8 some sensitive medicine

Does Smoke:

Phone Numbers: 02167542089

Signature

upload Edit

< - +

ID: 7

Pages

Personal Info 1
new page...

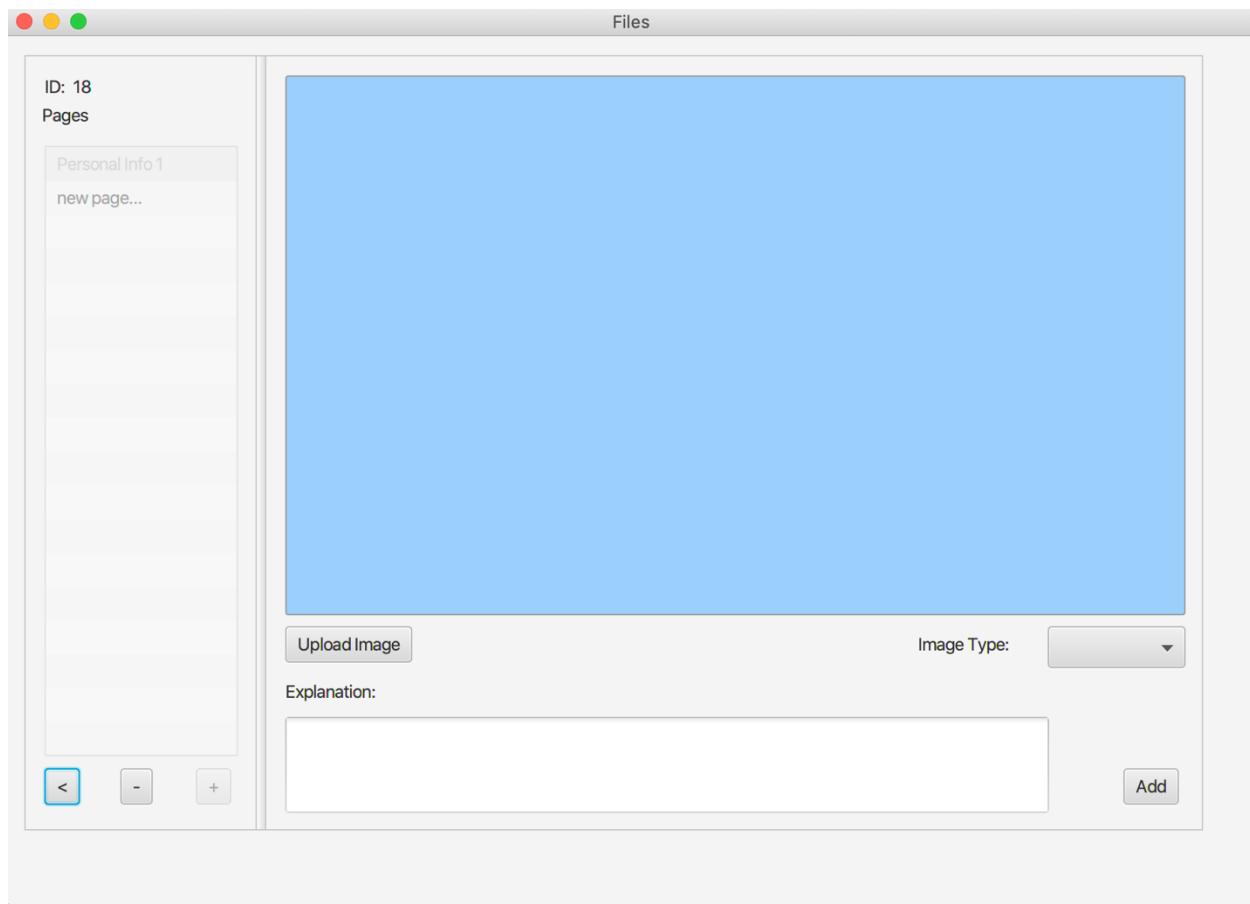
Treatment Summary:

Next Appointment Date:

Paid Amount Whole Amount

Appointment Time:
Select a referral time from the list below:

< - + Add



The screenshot shows a web-based application titled "booking" with a "weekly schedule" tab selected. The main title is "Current Weekly Schedule". A button for "Add New Schedule" is visible. Below the title is a grid for the days of the week: Saturday, Sunday, Monday, Tuesday, Wednesday, Thursday, and Friday. Each day has a 12x12 grid of light gray boxes. At the bottom, there is a section for entering dates: "From Date:" with a date input field, "To get Alternative Schedule Enter Date below To Date:" with a date input field, and a "get schedule" button.

weekly schedule timeline

Current Weekly Schedule

Add New Schedule

SATURDAY SUNDAY MONDAY TUESDAY WEDNESDAY THURSDAY FRIDAY

To get Alternative Schedule Enter Date below

From Date:

To Date:

booking

weekly schedule timeline

Enter Occupied Time:
Enter Date (YYYY-MM-DD)
Enter Begin Time (HH:MM:SS)
Enter End Time (HH:MM:SS)

Reason
Patient id (if it is a referral time)

Add Time to Schedule

Cancel Appointment

Selected Time:
Date:
Begin Time:
End Time:

06:00 07:00 08:00 09:00 10:00 11:00 12:00 13:00 14:00 15:
From time: (YYYY-MM-DD HH:MM:SS):
2021-01-02 00:00:00
Enter
To time: (YYYY-MM-DD HH:MM:SS):
2021-02-21 00:00:00

< >