ПРИЛОЖЕНИЕ А
(обязательное)
Программный код проекта

```
class Settings(BaseSettings):
    db_url: str
    cam_id: int
    cam_pos: bool
    cam_ip: str
    door_controller: str

print(sys.path)

settings = Settings(
    _env_file='.env',
)

db_url = 'postgresql://userP:mypass@192.168.126.130:5432/facedb'
engine = create_engine(
    db_url, pool_size=10, max_overflow=20
)
Session = sessionmaker(
    engine,
    autocommit=False,
    autoflush=False,
)
session = Session()
Base = declarative_base()
class facesDB(Base):
    __tablename__ = "faces"
    id = Column(Integer, primary_key=True)
    file = Column(BYTEA)
    face_data = Column(BYTEA)
    personal_id = Column(Integer)
class face_check(Base):
    __tablename__ = "in_out_date"
    time = Column(DateTime, default=datetime.datetime.now, primary_key=True)
    per_id = Column(Integer)
    cam_id = Column(Integer)
class undef_face(Base):
    __tablename__ = "undendified_faces"
    time = Column(DateTime, default=datetime.datetime.now, primary_key=True)
    cam_id = Column(Integer)
    file = Column(BYTEA)
def get_faces():
    #ans = session.query(facesDB).filter(facesDB.id == 0).all()
    ans = session.query(facesDB.personal_id, facesDB.face_data).all()

    answer_id = []
```

```
    answer_face = []
    for i in ans:
        bytes_converted = np.frombuffer( i[1], np.float64)
        answer_id.append(i[0])
        answer_face.append(bytes_converted)
    return answer_id, answer_face

def find_faces(known_faces, new_faces):
    for face_new in face_encodings:
        t1 = time.time()
        matches = fa_re.compare_faces(faces, face_new)
        if True in matches:
            print(f'id={ matches.index(True) }')
        else:
            print('unknown')
if __name__ == '__main__':
repeat = True
while repeat:
    try:
        sas = session.execute(text('SELECT 1'))
    except Exception as e:
        print("can't connect to data base")
        repeat=True
        time.sleep(5)
    else:
        repeat=False
repeat = True
while repeat:
    ids, faces = get_faces()
    if (len(faces)==0):
        repeat=True
        time.sleep(5)
    else:
        repeat=False
camera_i=session.query(camerasDB).filter(camerasDB.id==settings.cam_id).first()
cab_id=camera_i.cab_id
print(cab_id)
allowed_pers = session.query(cabinetsDB.pers_ids).filter(cabinetsDB.id==cab_id).all()
allowed_pers=allowed_pers[0][0]
if len(allowed_pers)==0:
    sys.exit()
video_capture = cv2.VideoCapture(settings.cam_ip)
repeat = True
while repeat:
    try:
        ret, frame = video_capture.read()
    except:
        repeat=True
```

```
        camera_i.status=False
        session.commit()
    else:
        camera_i.status=True
        session.commit()
        repeat=False

while True:

    ret, frame = video_capture.read()

    if process_this_frame:
        small_frame = cv2.resize(frame, (0, 0), fx=0.25, fy=0.25)

        rgb_small_frame = np.ascontiguousarray(small_frame[:, :, ::-1])

        face_locations = fa_re.face_locations(rgb_small_frame)
        if len(face_locations)==0:
            continue
        face_encodings = fa_re.face_encodings(rgb_small_frame, face_locations, model="small")
        t1 = time.time()
        for face_new in face_encodings:
            #time.sleep(0.5)
            matches = fa_re.compare_faces(faces, face_new)
            if True in matches:
                p_id=ids[ matches.index(True) ]
                print(f'id={ p_id }')
                face_add = face_check(
                    per_id=p_id,
                    cam_id =settings.cam_id
                )
                session.add(face_add)
                session.commit()
                if settings.cam_pos==True and p_id in allowed_pers:
                    #opendoor()
                    print("OPEN DOOR")
                    pass
            else:
                face_unden = undef_face(
                    cam_id =settings.cam_id,
                    file= cv2.imencode('.png', frame)[1].tobytes()
                )
                session.add(face_unden)
                session.commit()
                print('unknown')
        print(f"{time.time()-t1}")
```

ПРИЛОЖЕНИЕ Б
(обязательное)
Функции отчётов

```
CREATE OR REPLACE FUNCTION worker_day_visits
        (IN date_f date,IN per_in integer)
    returns TABLE (
        p_date timestamp,
        cab integer,
        pos boolean
    )
    AS
$$
DECLARE
BEGIN
    return query
        select
        public.in_out_date.time as p_date,
        public.cameras.cab_id as cab,
        public.cameras.in_pos as pos
        from public.in_out_date

        INNER JOIN public.cameras
        ON public.in_out_date.cam_id=public.cameras.id
        and DATE(public.in_out_date.time)=date_f
        and public.in_out_date.per_id=per_in
        ;
end;
$$
LANGUAGE 'plpgsql'
-----------------------------------------------------------------------
CREATE OR REPLACE FUNCTION worker_day_visits_pos
```

```
        (IN date_f date,IN per_in integer, IN pos_in boolean)
    returns TABLE (
        p_date timestamp,
        cab integer
    )
    AS
$$
DECLARE
BEGIN
    return query
        select
        public.in_out_date.time as p_date,
        public.cameras.cab_id as cab

        from public.in_out_date

        INNER JOIN public.cameras
        ON public.in_out_date.cam_id=public.cameras.id
        and DATE(public.in_out_date.time)=date_f
        and public.in_out_date.per_id=per_in
        and public.cameras.in_pos=pos_in
        ;
end;
$$
LANGUAGE 'plpgsql'
-----------------------------------------------------------------------
CREATE OR REPLACE FUNCTION cab_visits
        (IN date_f date,IN cab_in integer)
    returns TABLE (
        date_o timestamp,
        per integer,
```

```
            pos boolean
        )
        AS
$$
DECLARE
BEGIN
        return query
            select

            public.in_out_date.time as date_o,
            public.in_out_date.per_id as per,
            public.cameras.in_pos as pos

            from public.in_out_date

            INNER JOIN public.cameras
            ON public.in_out_date.cam_id=public.cameras.id
            and DATE(public.in_out_date.time)=date_f
            and public.cameras.cab_id=cab_in
            ;
end;
$$
LANGUAGE 'plpgsql'
------------------------------------------------------------------------
CREATE OR REPLACE FUNCTION cab_visits_pos
            (IN date_f date,IN cab_in integer, IN pos boolean)
        returns TABLE (
            date_o timestamp,
            per integer
        )
        AS
```

```
$$
DECLARE
BEGIN
	return query
		select

		public.in_out_date.time as date_o,
		public.in_out_date.per_id as per

		from public.in_out_date

		INNER JOIN public.cameras
		ON public.in_out_date.cam_id=public.cameras.id
		and DATE(public.in_out_date.time)=date_f
		and public.cameras.cab_id=cab_in
		and public.cameras.in_pos=pos
		;
end;
$$
LANGUAGE 'plpgsql'
-----------------------------------------------------------------------
CREATE OR REPLACE FUNCTION pass_visits
		(IN date_f date, IN pass integer,IN cab_in integer)
	returns TABLE (
		date_o timestamp,
		per integer,
		pas_bool boolean
	)
	AS
$$
DECLARE
```

```
BEGIN
	return query
		select

		public.in_out_date.time as date_o,
		public.in_out_date.per_id as per,
		public.cameras.in_pos as pas_bool

		from public.in_out_date

		INNER JOIN public.cameras
		ON public.in_out_date.cam_id=public.cameras.id
		and DATE(public.in_out_date.time)=date_f
		and public.cameras.pass_num=pass
		and public.cameras.cab_id=cab_in
		;
end;
$$
LANGUAGE 'plpgsql'
-----------------------------------------------------------------
CREATE OR REPLACE FUNCTION pass_visits_pos
		(IN date_f date, IN pass integer,
		 IN cab_in integer, IN pass_in boolean)
	returns TABLE (
		date_o timestamp,
		per integer,
		pas_bool boolean
	)
	AS
$$
DECLARE
```

```
BEGIN
	return query
		select

		public.in_out_date.time as date_o,
		public.in_out_date.per_id as per,
		public.cameras.in_pos as pas_bool

		from public.in_out_date

		INNER JOIN public.cameras
		ON public.in_out_date.cam_id=public.cameras.id
		and DATE(public.in_out_date.time)=date_f
		and public.cameras.pass_num=pass
		and public.cameras.cab_id=cab_in
		and public.cameras.in_pos=pass_in
		;
end;
$$
LANGUAGE 'plpgsql'
-----------------------------------------------------------------
```