LAMPIRAN

1. Server

A. *Config*

```
import os
import cloudinary
from sqlalchemy import
create_engine
from sqlalchemy.ext.declarative
import declarative_base
from sqlalchemy.orm import
sessionmaker
from dotenv import load_dotenv
load_dotenv()
cloudinary.config(
   cloud_name=os.getenv('CLOU
D_NAME'),
   api_key=os.getenv('API_KEY'),
   api_secret=os.getenv('API_SEC
RET')
)
SQLALCHEMY_DATABASE_URL
= os.getenv('DB_URL')
engine =
create_engine(SQLALCHEMY_D
ATABASE_URL)
SessionLocal =
sessionmaker(autocommit=False,
autoflush=False, bind=engine)
Base = declarative_base()
def get_db():


 db = SessionLocal()
  try:
     yield db
  finally:
     db.close()
```

B. *index.js*

```
import Vue from "vue";
import VueRouter from "vue-
router";
import store from "@/store";
Vue.use(VueRouter);
const isLogin = (to, from, next)
=> {
 const { login, role } =
store.state.authModule;
 try {
  if (to.name == "Login") {
   if (login) next(role);
  } else {
   if (!login) next("/");
   if (to.name.toLowerCase()
!= role) {
    next(role);
   }
  }
 } catch (error) {
  // console.log(error);c
 }
 next();
};
const routes = [
 {
  path: "/",
  name: "Login",
  beforeEnter: isLogin,
  component: () =>
import("../views/Login.vue"),
 },
 {
  path: "/logout",
  beforeEnter: (to, from, next)
=> {
   store.dispatch("authModule
/logout");
   next("/");
  },
 },
 {
  path: "/admin",
```

```
    component: () =>
import("../views/admin/index.vu
e"),
    beforeEnter: isLogin,
    children: [
      {
        path: "",
        name: "admin",
        component: () =>
import("../views/admin/dashboa
rd.vue"),
      },
      {
        path: "kandidat/",
        component: () =>
import("../views/admin/kandida
t.vue"),
      },
      {
        path: "pemilih/",
        component: () =>
import("../views/admin/pemilih.
vue"),
      },
      {
        path: "daftarvote/",
        component: () =>
import("../views/admin/daftarV
ote.vue"),
      },
      {
        path: "kotaksuara/",
        component: () =>
import("../views/admin/kotakSu
ara.vue"),
      },
      {
        path: "hitungcepat/",
        component: () =>
import("../views/admin/hitungC
epat.vue"),
      },
      {
        path: "laporan/",
        component: () =>
import("../views/admin/laporan.
vue"),
      },
      {
        path: "user/",
        component: () =>
import("../views/admin/user.vue
"),
      },
    ],
  },
  {
    path: "/petugas",
    component: () =>
import("../views/petugas/index.v
ue"),
    beforeEnter: isLogin,
    children: [
      {
        path: "",
        name: "petugas",
        component: () =>
import("../views/admin/dashboa
rd.vue"),
      },
      {
        path: "kandidat/",
        component: () =>
import("../views/admin/kandida
t.vue"),
      },

      {
        path: "pemilih/",
        component: () =>
import("../views/admin/pemilih.
vue"),
      },
      {
        path: "daftarvote/",
        component: () =>
import("../views/admin/daftarV
ote.vue"),
      },
      {
```

```
      path: "kotaksuara/",
      component: () =>
import("../views/admin/kotakSu
ara.vue"),
    },
    {
      path: "hitungcepat/",
      component: () =>
import("../views/admin/hitungC
epat.vue"),
    },
    {
      path: "laporan/",
      component: () =>
import("../views/admin/laporan.
vue"),
    },
  ],
 },
 {
  path: "/pemilih",
  component: () =>
import("../views/pemilih/index.v
ue"),
  beforeEnter: isLogin,
  children: [
   {
    path: "",
    name: "pemilih",
    meta: { text: "Home" },
    component: () =>
import("../views/pemilih/home.v
ue"),
   },
   {
    path: "vote/",
    name: "pemilih-vote",
    meta: { text: "Daftar Vote"
},
    component: () =>
import("../views/pemilih/vote.vu
e"),
   },
   {
    path: "akun/",
    name: "akun",
    meta: { text: "Akun" },
    component: () =>
import("../views/pemilih/akun.v
ue"),
   },
   {
    path: "facerecognition/",
    name: "face-recognition",
    meta: { text: "Daftar Face
Recognition" },
    component: () =>
import("../views/pemilih/faceRe
cognition.vue"),
   },
  ],
 },
];

const router = new VueRouter({
 // mode: "history",
 base: process.env.BASE_URL,
 routes,
});

export default router;
```

C. Face Recognition

(websocket.py)

```
import asyncio
import io
from app import app, lbph
import cv2
from fastapi import Depends,
WebSocket,
WebSocketDisconnect
from fastapi.encoders import
jsonable_encoder
from sqlalchemy.orm import
Session
from config.db import get_db
from models.pemilih import
Pemilihs
from services.token import
create_access_token
from schemas.authentication
import Token
import cloudinary.uploader
```

```
from PIL import Image
@app.websocket("/faceRecognition/{id_pemilih}")
async def websocket_endpoint(id_pemilih: int,websocket: WebSocket, db: Session = Depends(get_db)):
    await websocket.accept()
    totalDetect = 0
    maxDetect = 5
    currentDetect = None
    try:
        while True:
            progress = (totalDetect/maxDetect) * 100
            if progress > 100:
                return
            data = await websocket.receive_text()
            img = lbph.base64_cv2(data)
            imgBlur = cv2.Laplacian(img, cv2.CV_64F).var()
            response = {'detect': None, 'message': None, 'progress': 0}
            if imgBlur <= 100:
                response['message'] = 'Kamera blur. Gunakan kamera dengan resolusi lebih tinggi'
                await websocket.send_json(response)
                continue
            imgPredict = lbph.predict_image(img)

            if imgPredict['detect'] == True:
                if currentDetect == imgPredict['label']:
                    totalDetect += 1
                else:
                    totalDetect = 0
                currentDetect = imgPredict['label']
                response['message'] = 'Sedang mengenali wajah...'
                if currentDetect != id_pemilih:
                    imgPredict['detect'] = False
                    totalDetect = 0
                    response['message'] = 'Data wajah tidak sesuai'
            elif imgPredict['detect'] == False:
                response['message'] = 'Wajah tidak terdaftar'
                totalDetect = 0
            else:
                response['message'] = 'Tidak ada wajah terdeteksi'
                totalDetect = 0
            response['detect'] = imgPredict['detect']
            response['progress'] = progress
            if progress == 100:
                pemilih = db.query(Pemilihs).get(currentDetect)
                access_token = create_access_token(
                    data={"sub": pemilih.username})
                token = Token(access_token=access_token, token_type='bearer')
                response['token'] = jsonable_encoder(token)
                response['user'] = jsonable_encoder(pemilih)
                response['user']['role'] = 'pemilih'

            await websocket.send_json(response)
    except WebSocketDisconnect:
```

```
        pass
@
app.websocket("/regisFace/{id_
pemilih}")
async def
websocket_endpoint(websocket:
WebSocket, id_pemilih: int, db:
Session = Depends(get_db)):
    await websocket.accept()
    try:
        maxImg = 100
        file = 1
        while True:
            progress = (file /
maxImg) * 100
            if progress > 100:
                pemilih =
db.query(Pemilihs).get(id_pemi
lih)
                pemilih.face_recognit
ion = True
                db.commit()
                asyncio.create_task(l
bph.training_model())
                return
            data = await
websocket.receive_text()
            img =
lbph.base64_cv2(data)
            imgBlur =
cv2.Laplacian(img,
cv2.CV_64F).var()
            response = {"detect":
None, "message": "Tidak ada
wajah terdeteksi",
                        "progress":
progress}
            if imgBlur <= 100:
                response['message']
= 'Kamera blur. Gunakan
kamera dengan resolusi lebih
tinggi'
                await
websocket.send_json(response)
                continue
            imgDetect =
lbph.face_detection(img, False)
            if imgDetect[0] is not
None:
                name =
str(id_pemilih) + "." + str(file)
                imgColor =
cv2.cvtColor(imgDetect,
cv2.COLOR_BGR2RGB)
                image =
Image.fromarray(imgColor)
                imgByteArr =
image_to_byte_array(image)
                upload =
cloudinary.uploader.upload(
                    imgByteArr,
public_id=name,
folder='training',
invalidate=True, **{'async':
True})
                print(upload)
                response['detect'] =
True
                file += 1
                if file < 10:
                    response['message'
] = "Sedang merekam data
wajah..."
                elif file < 20:
                    response['message'
] = "Ubah posisi wajah dan
ekspresi wajah anda..."
                elif file <= 30:
                    response['message'
] = "Sedang merekam data
wajah..."
            await
websocket.send_json(response)
    except
WebSocketDisconnect:
        pass
def
image_to_byte_array(image:
Image):
    imgByteArr = io.BytesIO()
```

```
    image.save(imgByteArr,
format='PNG')
    imgByteArr =
imgByteArr.getvalue()
    return imgByteArr
```

D. Models

1. Daftar_vote.py

```
from sqlalchemy import JSON,
TIMESTAMP, Boolean,
Column, Date,
DateTime,  Integer, String,
Text, Time, func
from config.db import Base
class DaftarVotes(Base):
    __tablename__ =
"daftar_vote"
    id = Column(Integer,
primary_key=True,
index=True)
    nama = Column(String(100))
    keterangan = Column(Text)
    tanggal_mulai =
Column(Date())
    jam_mulai =
Column(Time(timezone=True))
    tanggal_selesai =
Column(Date())
    jam_selesai =
Column(Time(timezone=True))
    list_kandidat =
Column(JSON, default=[])
    list_pemilih =
Column(JSON, default=[])
    created_at =
Column(TIMESTAMP(timezone
=True),
server_default=func.now())
    updated_at =
Column(TIMESTAMP(timezone
=True), onupdate=func.now())
```

2. Kandidat.py

```
    from sqlalchemy import
    Column, Integer, String,
    Text
    from config.db import Base
    class Kandidats(Base):
        __tablename__ =
    "kandidat"
        id = Column(Integer,
    primary_key=True,
    index=True)
        nama =
    Column(String(100))
        foto = Column(Text)
        keterangan =
    Column(Text)
```

E. Cetak.py

```
from datetime import datetime,
date, time
import os
from fpdf import FPDF
from sqlalchemy.orm import
Session
from models.daftar_vote import
DaftarVotes
from models.kandidat import
Kandidats
from models.pemilih import
Pemilihs
import urllib.request as req
def cetak_kandidat(db:
Session):
    path = 'assets/kandidat.pdf'
    cloud_name =
os.getenv('CLOUD_NAME')
    pdf = FPDF()
    pdf.add_page(orientation="l
andscape", format="LEGAL")

    TABLE_COL_NAMES =
("No", "Foto", "Nama",
"Keterangan")
    kandidats =
db.query(Kandidats).all()
    TABLE_DATA =
((str(idx+1), kandidat.foto,
kandidat.nama,
kandidat.keterangan)
            for idx, kandidat in
enumerate(kandidats))
```

```
    pdf.set_font("Times",
size=16)
    line_height = pdf.font_size *
2
    TABLE_COL_WIDTH =
[10.0, 50.0, 80.0, 200.0]
    render_table_header(pdf,
TABLE_COL_NAMES,TABLE_
COL_WIDTH, line_height)
    for row in TABLE_DATA:
        if
pdf.will_page_break(line_heigh
t):
            render_table_header(pd
f,
TABLE_COL_NAMES,TABLE_
COL_WIDTH, line_height)
        y = pdf.get_y()
        lh =
max([(pdf.get_string_width(dat
a)/TABLE_COL_WIDTH[idx])
*8 for idx, data in
enumerate(row)]);
        for idx, data in
enumerate(row):
            if idx == 1:
                pdf.cell(TABLE_COL
_WIDTH[idx], lh,  border=1)
                y = pdf.get_y()
                x = pdf.get_x()
                pdf.image(
                    f"https://res.cloudi
nary.com/{cloud_name}/image/
upload/q_auto:low,w_100,h_10
0/{data}", h=20, w=20, x=35,
y=y+(lh/6))
                pdf.set_y(y)
                pdf.set_x(x)

            elif idx ==
3:
                pdf.multi_cell(TABLE
_COL_WIDTH[idx], lh,
                    data, border=1,
align="C",
max_line_height=6,ln=0 if
len(row) != idx+1 else 1)
            else:
                pdf.cell(TABLE_COL
_WIDTH[idx], lh, data,
border=1, align="C")
    pdf.output(path)
    return path
def cetak_pemilih(db: Session):
    path = 'assets/pemilih.pdf'
    pdf = FPDF()
    pdf.add_page()
    TABLE_COL_NAMES =
("NO", "NAMA", "NIK",
"USERNAME", "ALAMAT",
"STATUS")
    pemilihs =
db.query(Pemilihs).all()
    def format_status(status):
        if status is True:
            return 'Aktif'
        elif status is False:
            return 'Tidak Aktif'
        else:
            return 'Belum
dikonfirmasi'
    TABLE_DATA =
((str(idx+1), kandidat.nama,
kandidat.nik,
kandidat.username,
kandidat.alamat,
format_status(kandidat.status))
                for idx, kandidat in
enumerate(pemilihs))
    pdf.set_font("Times",
size=10)
    line_height = pdf.font_size *
2
    col_width = pdf.epw / 6

    TABLE_COL_WIDTH =
[col_width + ((col_width-10) /
5) for i in range(5)]
    TABLE_COL_WIDTH.insert(
0, 10.0)
```

```
        render_table_header(pdf,
TABLE_COL_NAMES,TABLE_
COL_WIDTH, line_height)
    for row in TABLE_DATA:
        if
pdf.will_page_break(line_heigh
t):
            render_table_header(pd
f,
TABLE_COL_NAMES,TABLE_
COL_WIDTH, line_height)
        for idx, data in
enumerate(row):
            pdf.cell(TABLE_COL_
WIDTH[idx], line_height,
                    data, border=1,
align="C")
        pdf.ln(line_height)
    pdf.output(path)
    return path
def cetak_daftar_vote(db:
Session):
    path =
'assets/daftar_vote.pdf'
    pdf = FPDF()
    pdf.add_page(orientation="l
andscape", format="LEGAL")
    TABLE_COL_NAMES =
("NO", "NAMA VOTE",
"KETERANGAN", "TANGGAL
MULAI",
                "TANGGAL
SELESAI", "JUMLAH
KANDIDAT", "JUMLAH
PEMILIH", "STATUS")
    daftar_votes =
db.query(DaftarVotes).all()
    def format_tanggal(tanggal:
date, jam: time):
        return
datetime.combine(tanggal,
jam).strftime("%m/%d/%Y,
%H:%M:%S")

    def
format_status(tanggal_mulai:
datetime, tanggal_selesai:
datetime):
        now = datetime.now()
        if now < tanggal_mulai:
            return 'Belum Aktif'
        elif tanggal_mulai < now
< tanggal_selesai:
            return 'Aktif'
        elif now >
tanggal_selesai:
            return 'Selesai'
    TABLE_DATA =
((str(idx+1), daftar_vote.nama,
daftar_vote.keterangan,
format_tanggal(daftar_vote.tan
ggal_mulai,
daftar_vote.jam_mulai),
format_tanggal(daftar_vote.tan
ggal_selesai,
daftar_vote.jam_selesai),
str(len(daftar_vote.list_kandida
t)),
str(len(daftar_vote.list_pemilih)
),
format_status(datetime.combine
(daftar_vote.tanggal_mulai,
daftar_vote.jam_mulai),
datetime.combine(daftar_vote.t
anggal_selesai,
daftar_vote.jam_selesai)))
                for idx, daftar_vote
in enumerate(daftar_votes))
    pdf.set_font("Times",
size=12)
    line_height = pdf.font_size *
2
    col_width = pdf.epw /
len(TABLE_COL_NAMES)
    TABLE_COL_WIDTH =
[col_width + (-(10 - col_width)
/ (len(TABLE_COL_NAMES)-
1))
                for i in
range(len(TABLE_COL_NAME
S)-1)]
```

```
    TABLE_COL_WIDTH.insert(
0, 10.0)
    render_table_header(pdf,
TABLE_COL_NAMES,TABLE_
COL_WIDTH, line_height)
    for row in TABLE_DATA:
        if
pdf.will_page_break(line_heigh
t):
            render_table_header(pd
f,
TABLE_COL_NAMES,TABLE_
COL_WIDTH, line_height)
        y = pdf.get_y()
        lh =
max([(pdf.get_string_width(dat
a)/TABLE_COL_WIDTH[idx])
*8 for idx, data in
enumerate(row)]);
            for idx, data in
enumerate(row):
            pdf.set_xy(pdf.get_x(),
y)
            pdf.multi_cell(TABLE_
COL_WIDTH[idx], lh,
                    data, border=1,
align="C",
max_line_height=6,ln=0 if
len(row) != idx+1 else 1)
        # pdf.ln(line_height)
    pdf.output(path)
    return path
def render_table_header(pdf,
TABLE_COL_NAMES,TABLE_
COL_WIDTH, line_height):
        pdf.set_font(style="B")
        for idx, col_name in
enumerate(TABLE_COL_NAM
ES):
            pdf.cell(TABLE_COL_
WIDTH[idx], line_height,
col_name,
                    border=1,
align='C')
        pdf.ln(line_height)
        pdf.set_font(style="")
```

F. Daftar_Vote.py

```
from datetime import datetime
import json
from fastapi import status,
HTTPException
from fastapi.encoders import
jsonable_encoder
from sqlalchemy.orm import
Session
from sqlalchemy.exc import
SQLAlchemyError
from models.daftar_vote import
DaftarVotes
from models.kandidat import
Kandidats
from models.pemilih import
Pemilihs
from models.user import Users
from schemas.daftar_vote
import DaftarVoteCreate,
DaftarVotePemilih,
DaftarVoteResponse
from schemas.kandidat import
KandidatHitungCepat,
KandidatVoteCreate
from schemas.pemilih import
PemilihKotakSuara,
PemilihVoteCreate
from services.error_handling
import add_or_edit_exception

def get_all(db: Session):
    return
db.query(DaftarVotes).all()

def create(req:
DaftarVoteCreate, db:
Session):
    try:
        new = DaftarVotes(
            nama=req.nama,
            keterangan=req.keteran
gan,
            tanggal_mulai=req.tang
gal_mulai,
```

```
            tanggal_selesai=req.tanggal_selesai,
            jam_mulai=req.jam_mulai,
            jam_selesai=req.jam_selesai,
        )
        db.add(new)
        db.commit()
        db.refresh(new)

        return DaftarVoteResponse(message="Daftar Vote berhasil ditambahkan", item=new)

    except SQLAlchemyError as e:
        add_or_edit_exception(e)

async def update(id: int, req: DaftarVoteCreate, db: Session):
    try:
        user = cek_daftar_vote(id, db)
        user.update(req.dict())
        db.commit()

        return DaftarVoteResponse(message="Daftar Vote berhasil diubah")
    except SQLAlchemyError as e:
        add_or_edit_exception(e)

async def delete(id: int,  db: Session):
    try:
        user = cek_daftar_vote(id, db)

        user.delete(synchronize_session=False)
        db.commit()
        return DaftarVoteResponse(message="Daftar Vote berhasil dihapus")
    except SQLAlchemyError as e:
        raise HTTPException(
            status_code=status.HTTP_400_BAD_REQUEST, detail=str(e.__dict__['orig']))

def get_list(id: int, db: Session):
    list = db.query(DaftarVotes.list_kandidat, DaftarVotes.list_pemilih).filter(
        DaftarVotes.id == id).first()

    if not list:
        raise HTTPException(status_code=status.HTTP_404_NOT_FOUND,
                            detail="ID Daftar Vote tidak ditemukan")
    return list

def get_kotak_suara(id: int, db: Session):
    try:
        list = db.query(DaftarVotes.list_pemilih).filter(
            DaftarVotes.id == id).first()

        if not list:
            raise HTTPException(status_code=status.HTTP_404_NOT_FOUND,
                                detail="ID Daftar Vote tidak ditemukan")
```

```
        filter = sorted([pemilih for
pemilih in list.list_pemilih if
pemilih['waktu']],
                        key=lambda v:
v['id'],)

        ids = (int(pemilih['id']) for
pemilih in filter)

        pemilih =
db.query(Pemilihs.nama,
Pemilihs.username).filter(
            Pemilihs.id.in_(ids)).all(
)

        data:
list[PemilihKotakSuara] = []

        if pemilih:
            for i in
range(len(filter)):
                fil = filter[i]
                pem = pemilih[i]
                data.append(Pemilih
KotakSuara(id=fil['id'],
vote_nomor=fil['vote_nomor'],
                            waktu=fil['wa
ktu'], nama=pem['nama'],
username=pem['username']))

        return data
    except SQLAlchemyError as
e:
        raise HTTPException(
            status_code=status.HTT
P_400_BAD_REQUEST,
detail=str(e.__dict__['orig']))

def get_hitung_cepat(id: int,
db: Session):
    try:
        list =
db.query(DaftarVotes.list_kandi
dat,
DaftarVotes.list_pemilih).filter(
            DaftarVotes.id ==
id).first()

        if not list:
            raise
HTTPException(status_code=st
atus.HTTP_404_NOT_FOUND
,
                          detail="ID
Daftar Vote tidak ditemukan")

        ids = (kandidat['id'] for
kandidat in list.list_kandidat)

        kandidat =
db.query(Kandidats).filter(Kan
didats.id.in_(ids)).all()

        # menghitung total vote
        vote = {}
        for pemilih in
list.list_pemilih:
            if pemilih['vote_nomor']
in vote:
                vote[(pemilih['vote_n
omor'])] += 1
            else:
                vote[pemilih['vote_no
mor']] = 1

        data:
list[KandidatHitungCepat] = []

        for i, list_kandidat in
enumerate(list.list_kandidat):
            data.append(KandidatH
itungCepat(id=list_kandidat['id
'], nama=kandidat[i].nama,
nomor=list_kandidat['nomor'],
                        keterangan=kan
didat[i].keterangan,
foto=kandidat[i].foto,
jumlah=vote[str(
                            list_kandidat['
nomor'])]
```

```
            if
str(list_kandidat['nomor']) in
vote else 0))

        return data

    except SQLAlchemyError as
e:
        raise HTTPException(
            status_code=status.HTT
P_400_BAD_REQUEST,
detail=str(e.__dict__['orig']))

def add_list_kandidat(id: int,
kandidats:
list[KandidatVoteCreate], db:
Session):
    try:
        daftar_vote =
db.query(DaftarVotes).get(id)

        if not daftar_vote:
            raise
HTTPException(status_code=st
atus.HTTP_404_NOT_FOUND
,
                          detail="ID
Daftar Vote tidak ditemukan")

        daftar_vote.list_kandidat
= jsonable_encoder(kandidats)

        db.commit()

        return
DaftarVoteResponse(message=
"Daftar Kandidat berhasil
disimpan")
    except SQLAlchemyError as
e:
        raise HTTPException(
            status_code=status.HTT
P_400_BAD_REQUEST,
detail=str(e.__dict__['orig']))

def add_list_pemilih(id: int,
pemilihs:
list[PemilihVoteCreate], db:
Session):
    try:
        # raise HTTPException(
        #     status_code=status.H
TTP_400_BAD_REQUEST,
detail='Daftar Vote
Aktif/Berjalan tidak dapat
diubah')

        daftar_vote =
db.query(DaftarVotes).get(id)

        if not daftar_vote:
            raise
HTTPException(status_code=st
atus.HTTP_404_NOT_FOUND
,
                          detail="ID
Daftar Vote tidak ditemukan")

        daftar_vote.list_pemilih =
jsonable_encoder(pemilihs)

        db.commit()

        return
DaftarVoteResponse(message=
"Daftar Pemilih berhasil
disimpan")
    except SQLAlchemyError as
e:
        raise HTTPException(
            status_code=status.HTT
P_400_BAD_REQUEST,
detail=str(e.__dict__['orig']))

def add_kandidat(id: int,
kandidat: KandidatVoteCreate,
db: Session):
    daftar_vote =
db.query(DaftarVotes).get(id)
```

```
    list_kandidat =
daftar_vote.list_kandidat.copy()

    # cek jika nomor urut telah
ada
    cek = next((i for i, d in
enumerate(
        list_kandidat) if d['nomor']
== kandidat.nomor), None)

    if cek is not None:
        raise
HTTPException(status_code=st
atus.HTTP_400_BAD_REQUE
ST,
                      detail='Nomor
urut kandidat telah digunakan')

    list_kandidat.append(kandid
at.dict())

    daftar_vote.list_kandidat =
list_kandidat

    db.commit()

    return
DaftarVoteResponse(message=
"Kandidat berhasil
ditambahkan")

def solve(key, val, lis):
    return next((i for i, d in
enumerate(lis) if d[key] ==
val), None)

def cek_daftar_vote(id: int, db:
Session):
    daftar_vote =
db.query(DaftarVotes).filter(
        DaftarVotes.id == id)

    if not daftar_vote.first():
        raise
HTTPException(status_code=st
atus.HTTP_404_NOT_FOUND
,
                      detail="ID
Daftar Vote tidak ditemukan")

    return daftar_vote

def
get_daftar_vote_pemilih(id_pe
milih: int, db: Session):
    query = db.execute(
        "SELECT *,
JSON_UNQUOTE(JSON_EXT
RACT(list_pemilih,
REPLACE(JSON_UNQUOTE(J
SON_SEARCH(list_pemilih,
'one', ':id', NULL, '$[*].id')),
'id', 'vote_nomor'))) as
vote_nomor FROM daftar_vote
WHERE
JSON_SEARCH(list_pemilih,
'one', :id, NULL, '$[*].id') IS
NOT NULL", {'id':
id_pemilih}).all()

    daftar_vote:
list[DaftarVotePemilih] = []

    for daftar in query:
        waktu_mulai =
datetime.strptime(
            f"{daftar.tanggal_mulai
} {daftar.jam_mulai}", "%Y-
%m-%d %H:%M:%S")
        waktu_selesai =
datetime.strptime(
            f"{daftar.tanggal_selesa
i} {daftar.jam_selesai}", "%Y-
%m-%d %H:%M:%S")

        list_kandidat =
[KandidatVoteCreate(
            id=int(kandidat['id']),
nomor=int(kandidat['nomor']))
for kandidat in
```

```
json.loads(daftar.list_kandidat)
]

        telah_memilih =
daftar.vote_nomor != '-1'

        daftar_vote.append(Daftar
VotePemilih(id=daftar.id,
nama=daftar.nama,
keterangan=daftar.keterangan,
tanggal_mulai=waktu_mulai.da
te(),
jam_mulai=waktu_mulai.time()
,
                                tan
ggal_selesai=waktu_selesai.dat
e(),
jam_selesai=waktu_selesai.tim
e(),
list_kandidat=list_kandidat,
telah_memilih=telah_memilih,
vote_nomor=daftar.vote_nomor
))

    return daftar_vote
```

G. *Daftar_Kandidat.py*

```
import uuid
from fastapi import
BackgroundTasks, File,
UploadFile, status,
HTTPException
from sqlalchemy.orm import
Session
from sqlalchemy.exc import
SQLAlchemyError
from models.kandidat import
Kandidats
from schemas.kandidat import
KandidatCreate,
KandidatResponse
import cloudinary
import cloudinary.uploader
import cloudinary.api
def get_all(db: Session):
    return
db.query(Kandidats).all()
def get_kandidat(id: int, db:
Session):
    kandidat =
db.query(Kandidats).get(id)
    return kandidat
async def create(req:
KandidatCreate, db: Session,
file: UploadFile):
    try:
        if file.content_type not in
['image/jpeg', 'image/jpg',
'image/png']:
            raise
HTTPException(status_code=st
atus.HTTP_406_NOT_ACCEP
TABLE,
                                detail="Ha
nya file foto yang dapat
diupload")
        file_name =
f'{uuid.uuid4().hex}'
        foto = await file.read()
        upload =
cloudinary.uploader.upload(
            foto,
public_id=file_name,
folder="kandidat",
invalidate=True)
        print(upload['url'])
        url =
upload['url'].split('upload/')[1]
        new_kandidat =
Kandidats(
            nama=req.nama,
foto=url,
keterangan=req.keterangan)
        db.add(new_kandidat)
```

```
        db.commit()
        db.refresh(new_kandidat)
        return
KandidatResponse(message="
Kandidat berhasil
ditambahkan",
item=new_kandidat)
    except SQLAlchemyError as
e:
        raise HTTPException(
            status_code=status.HTT
P_400_BAD_REQUEST,
detail=str(e.__dict__['orig']))
async def update(id: int, req:
KandidatCreate, db: Session,
file: UploadFile = File(None)):
    kandidat = cek_kandidat(id,
db)
    if file:
        foto = await file.read()
        url = kandidat.first().foto
        public_id = url.split('/')[-
1].replace('.jpg', '')
        upload =
cloudinary.uploader.upload(
            foto,
public_id=public_id,
folder="kandidat",
invalidate=True)
        url =
upload['url'].split('upload/')[-1]
        kandidat.update({'foto':
url})
    kandidat.update(req.dict())
    db.commit()
    return
KandidatResponse(message="
Kandidat berhasil diubah",
item=kandidat.first())
async def delete(id: int,
bg_task: BackgroundTasks, db:
Session):
    try:
        kandidat =
cek_kandidat(id, db)
        file_name =
kandidat.first().foto
        kandidat.delete(synchroniz
e_session=False)
        db.commit()
        bg_task.add_task(hapus_fi
le, file_name)
        return
KandidatResponse(message="
Kandidat berhasil dihapus")
    except SQLAlchemyError as
e:
        raise HTTPException(
            status_code=status.HTT
P_400_BAD_REQUEST,
detail=str(e.__dict__['orig']))
def hapus_file(public_id: str):
    cloudinary.uploader.destroy(
"kandidat/"+public_id)
    print(public_id + "
dihapus...")
def cek_kandidat(id: int, db:
Session):
    kandidat =
db.query(Kandidats).filter(Kan
didats.id == id)
    if not kandidat.first():
        raise
HTTPException(status_code=st
atus.HTTP_404_NOT_FOUND
,
                    detail="ID
Kandidat tidak ditemukan")
    return kandidat
```

H. Pemilih.py

```
from fastapi import
BackgroundTasks, status,
HTTPException
from sqlalchemy.orm import
Session
from sqlalchemy.exc import
SQLAlchemyError
from models.pemilih import
Pemilihs
from models.user import Users
from schemas.pemilih import
PemilihCreate,
PemilihResponse, PemilihVerif
from services.error_handling
import add_or_edit_exception
from services.sms_gateway
import kirim_sms
import cloudinary
import os
save_path =
'assets/foto/kandidat/'
def get_all(db: Session):
    return
db.query(Pemilihs).all()
def get_pemilih(id: int, db:
Session):
    try:
        pemilih = cek_pemilih(id,
db).first()
        return pemilih
    except SQLAlchemyError as
e:
        add_or_edit_exception(e)
async def create(req:
PemilihCreate,bg_task:
BackgroundTasks, db: Session):
    try:
        cek_username(req.userna
me, db)
        new = Pemilihs(
            nik=req.nik,
nama=req.nama,
username=req.username,
password=req.password,
alamat=req.alamat,
status=req.status,
telpon=req.telpon)
        db.add(new)
        db.commit()
        db.refresh(new)

        pesan = f"Akun anda telah
didaftarkan untuk melakukan
online voting. Silahkan login
menggunakan username :
{req.username}, password :
{req.password} di
{os.getenv('CLIENT_URL')}."
        bg_task.add_task(kirim_s
ms, req.telpon, pesan)
        return
PemilihResponse(message="Pe
milih berhasil ditambahkan",
item=new)
    except SQLAlchemyError as
e:
        add_or_edit_exception(e)
async def update(id: int, req:
PemilihCreate, db: Session):
    try:
        pemilih = cek_pemilih(id,
db)
        cek_username(req.userna
me, db)
        pemilih.update(req.dict())
        db.commit()
        return
PemilihResponse(message="Pe
milih berhasil diubah")
    except SQLAlchemyError as
e:
        add_or_edit_exception(e)
async def delete(id: int,
bg_task: BackgroundTasks,  db:
Session):
    try:
        pemilih = cek_pemilih(id,
db)
        pemilih.delete(synchronize
_session=False)
        db.commit()
```

```
        bg_task.add_task(hapus_fi
le, id)
        return
PemilihResponse(message="Pe
milih berhasil dihapus")
    except SQLAlchemyError as
e:
        raise HTTPException(
            status_code=status.HTT
P_400_BAD_REQUEST,
detail=str(e.__dict__['orig']))

async def verifikasi(id: int, req:
PemilihVerif, db: Session):
    try:
        pemilih =
db.query(Pemilihs).get(id)
        if not pemilih:
            raise
HTTPException(status_code=st
atus.HTTP_404_NOT_FOUND
,
                          detail="ID
Pemilih tidak ditemukan")
        pemilih.status = req.status
        db.commit()
        return
PemilihResponse(message="Pe
milih berhasil
di{}".format("terima" if
req.status else "tolak"))
    except SQLAlchemyError as
e:
        add_or_edit_exception(e)
def cek_pemilih(id: int, db:
Session):
    pemilih =
db.query(Pemilihs).filter(
        Pemilihs.id == id)
    if not pemilih.first():
        raise
HTTPException(status_code=st
atus.HTTP_404_NOT_FOUND
,
                        detail="ID
Pemilih tidak ditemukan")
    return pemilih
def hapus_file(id: str):
    public_ids = []
    for i in range(1,101):
        public_ids.append('trainin
g/{}.{}'.format(id, i))
    delete =
cloudinary.api.delete_resources
(public_ids)
def cek_username(username:
str, db: Session):
    cek =
db.query(Users.id).filter(Users.
username ==
username).count()

    if cek > 0:
        raise HTTPException(
            status_code=status.HTT
P_400_BAD_REQUEST,
detail="Username telah
digunakan")
```

I. *Total.py*

```
from datetime import datetime
from fastapi import
HTTPException, status
from sqlalchemy import between,
func
from sqlalchemy.orm import
Session
from sqlalchemy.exc import
SQLAlchemyError
from models.daftar_vote import
DaftarVotes
from models.kandidat import
Kandidats
from models.pemilih import
Pemilihs
from schemas.total import
Perolehan, Total, TotalKandidat,
TotalPerolehanSuara
import pandas as pd
import numpy as np
def get_total_data(db: Session):
    try:
        total_kandidat =
db.query(Kandidats.id).count()
```

```
        total_pemilih =
db.query(Pemilihs.id).filter(
            Pemilihs.status ==
True).count()
        now = datetime.now()
        # total_vote_aktif =
db.query(DaftarVotes.tanggal_mul
ai, DaftarVotes.jam_mulai).filter(
        #     func.timestamp(DaftarVo
tes.tanggal_mulai,
DaftarVotes.jam_mulai) <= now,
func.timestamp(DaftarVotes.tangg
al_selesai,
DaftarVotes.jam_selesai) >=
now).count()
        total_vote_aktif =
db.query(DaftarVotes.nama).filter(
between(now,
DaftarVotes.tanggal_mulai+Dafta
rVotes.jam_mulai,
DaftarVotes.tanggal_selesai+Daft
arVotes.jam_selesai)).count()

        return
Total(kandidat=total_kandidat,
pemilih=total_pemilih,
vote_aktif=total_vote_aktif)
    except SQLAlchemyError as e:
        print(e)
        raise HTTPException(
            status_code=status.HTTP_
400_BAD_REQUEST,
detail=str(e.__dict__['orig']))
def
get_perolehan_suara(id_daftarVot
e: int, db: Session):
    try:
        list =
db.query(DaftarVotes).get(id_daft
arVote)
        if not list:
            raise
HTTPException(status_code=statu
s.HTTP_404_NOT_FOUND,
                                detail="ID
Daftar Vote tidak ditemukan")
        ids = (kandidat['id'] for
kandidat in list.list_kandidat)
        kandidat_db =
db.query(Kandidats.id,
Kandidats.nama).filter(
            Kandidats.id.in_(ids)).all()
        telah_memilih = len([
            pemilih for pemilih in
list.list_pemilih if
pemilih['waktu']])
        belum_memilih =
len(list.list_pemilih) -
telah_memilih
        list_pemilih =
sorted(list.list_pemilih,
                               key=lambda v:
(v['waktu'] is not None,
v['waktu']))
        list_pemilih_map =
{pemilih['waktu']:
int(pemilih['vote_nomor'])
                            for pemilih in
list_pemilih if pemilih['waktu']}
        list_waktu: list[datetime] =
[datetime.fromisoformat(
            s['waktu']) for s in
list_pemilih if s['waktu']]
        waktu_mulai =
datetime.combine(
            list.tanggal_mulai,
list.jam_mulai)
        waktu_selesai =
datetime.combine(
            list.tanggal_selesai,
list.jam_selesai)

        waktu_selesai =
waktu_selesai if waktu_selesai <
datetime.now(
        ) else
datetime.now().replace(microseco
nd=0)
        print(waktu_mulai)
        print(waktu_selesai)
        # membuat daftar list waktu
        for i in range(11):
            waktu =
random_date(start=waktu_mulai,
                                end=waktu_sel
esai, position=i / 10)
            print(waktu)
```

```
        list_waktu.append(waktu.to
_pydatetime())
    list_waktu =
sorted(list_waktu, key=lambda v:
v)
    kandidats:
list[TotalKandidat] = []
    # membuat daftar list
kandidat
    for list_kandidat in
list.list_kandidat:
        perolehans:
list[Perolehan] = []
        total = 0
        for waktu in list_waktu:
            if str(waktu) in
list_pemilih_map and
list_pemilih_map[str(waktu)] ==
list_kandidat['nomor']:
                total += 1
            perolehans.append(Pero
lehan(waktu=waktu, total=total))
        nama = next(
            (kandidat['nama'] for
kandidat in kandidat_db if
kandidat['id'] ==
list_kandidat['id']), None)
        kandidat = TotalKandidat(
            nama=nama,
perolehan=perolehans)
        kandidats.append(kandidat
)

    return
TotalPerolehanSuara(telah_memil
ih=telah_memilih,
belum_memilih=belum_memilih,
kandidat=kandidats)
  except SQLAlchemyError as e:
    raise HTTPException(
        status_code=status.HTTP_
400_BAD_REQUEST,
detail=str(e.__dict__['orig']))
def random_date(start, end,
position=None):
  try:
    start, end = pd.Timestamp(
        start.strftime("%m/%d/%y
%I:%M%p")),
pd.Timestamp(end.strftime("%m/
%d/%y %I:%M%p"))
    delta = (end -
start).total_seconds()
    if position is None:
        offset =
np.random.uniform(0., delta)
    else:
        offset = position * delta
    offset =
pd.offsets.Second(offset)
    t = start + offset
    return t
  except:
    pass
```

J. *Vote.py*

```
import json
from datetime import datetime
from fastapi import
HTTPException, status
from fastapi.encoders import
jsonable_encoder
from sqlalchemy import func,
literal
from sqlalchemy.orm import
Session
from sqlalchemy.exc import
SQLAlchemyError

from models.daftar_vote import
DaftarVotes
from schemas.daftar_vote
import DaftarVoteResponse
def add_voting(id_daftarVote:
int, id_pemilih: int,
vote_nomor: int, db: Session):
  try:
    daftar_vote =
db.query(DaftarVotes).get(id_d
aftarVote)
    if not daftar_vote:
        raise
HTTPException(status_code=st
atus.HTTP_404_NOT_FOUND
,
                    detail="ID
Daftar Vote tidak ditemukan")
```

```
        result = db.execute(
            "SELECT
JSON_SET(list_pemilih,
REPLACE(JSON_UNQUOTE(J
SON_SEARCH(list_pemilih,
'one', :id_pemilih, NULL,
'$[*].id')), '.id', ''),
JSON_OBJECT('id',
':id_pemilih','vote_nomor',
':vote_nomor', 'waktu', :waktu))
as update_list_pemilih,
JSON_LENGTH(list_kandidat)
as total_kandidat FROM
daftar_vote where id = :id",
{'id': id_daftarVote,
'id_pemilih': id_pemilih,
'vote_nomor': vote_nomor,
'waktu':
datetime.now().replace(microse
cond=0)}).first()
        if not result[0]:
            raise
HTTPException(status_code=st
atus.HTTP_404_NOT_FOUND
,
    detail="ID Pemilih tidak
terdaftar pada Daftar Vote")
        if vote_nomor >
result.total_kandidat:
            raise
HTTPException(status_code=st
atus.HTTP_404_NOT_FOUND
,
            detail="Nomor Kandidat
tidak terdaftar pada Daftar
Vote")
        daftar_vote.list_pemilih =
json.loads(result.update_list_pe
milih)
        db.commit()
        return
DaftarVoteResponse(message=
"Voting berhasil disimpan")
    except SQLAlchemyError as
e:
        raise HTTPException(
            status_code=status.HTT
P_400_BAD_REQUEST,
detail=str(e.__dict__['orig']))
```

K. Eror Heandling.py

```
from fastapi import
HTTPException, status
from sqlalchemy.exc import
SQLAlchemyError
def
add_or_edit_exception(error:
SQLAlchemyError):
    err_msg = error.args[0]
    print(err_msg)
    if 'Duplicate entry' in
err_msg:
        field = ''
        if 'username' in err_msg:
            field = 'Username'
        elif 'nik' in err_msg:
            field = 'NIK'
        raise HTTPException(
            status_code=status.HTT
P_400_BAD_REQUEST,
detail=f'{field} telah
digunakan')
    raise HTTPException(
        status_code=status.HTTP_
400_BAD_REQUEST,
detail='Data gagal disimpan,
terjadi masalah diserver')
```

L. LBPH.py

```
import asyncio
import base64
import os
import cv2
import numpy as np
import cloudinary
import urllib.request as req
from alive_progress import
alive_bar
class LBPH():
    def __init__(self):
        # # self.model =
cv2.face_EigenFaceRecognizer.
create()
```

```
        # # self.model =
cv2.face_FisherFaceRecognizer
.create()
        self.model =
cv2.face_LBPHFaceRecognizer
.create()
        try:
            self.model.read("assets/l
bph_model.yml")
        except:
            asyncio.create_task(self.
training_model())
            async def
training_model(self):
        faces, labels = await
self.prepare_data()
        if len(faces) == 0:
            return
        self.model.train(faces,
np.array(labels))
        self.model.save(filename=
"assets/lbph_model.yml")
    async def prepare_data(self):
        labels = []
        faces = []
        resources = []
        next_cursor=None
        while(next_cursor is not
False):
            try:
                results =
cloudinary.api.resources(
                    prefix="training",
type="upload",
max_results=500,next_cursor=
next_cursor)

                resources.extend(resu
lts['resources'])
                next_cursor =
results['next_cursor']
            except:
                next_cursor =
False
        total = len(resources)
        print("Total {}
foto".format(total))
        with alive_bar(total,
ctrl_c=True) as bar:
            for result in resources:
                url =
result['url'].replace('http','https'
)
                print("download
image " + url)
                imgPath = 'assets/' +
\
                    result['public_id'].s
plit('/')[-1]+"."+result['format']
                req.urlretrieve(url,
imgPath)
                imgTraining =
cv2.imread(imgPath)
                face =
self.face_detection(imgTraining
)
                label =
int(os.path.split(imgPath)[-
1].split('.')[0])
                if face[0] is not None:
                    faces.append(face)
                    labels.append(label
)
                bar()
        print('Training Done')
        print('Total Wajah
{}'.format(len(np.unique(labels)
)))
        return faces, labels
    def face_detection(self,
image,
return_img_gray=True):
        try:
            image_gray =
cv2.cvtColor(image,
cv2.COLOR_BGR2GRAY)
            haar_classifier =
cv2.CascadeClassifier(
                'assets/haarcascade_f
rontalface_default.xml')
```

```
        # haar_classifier =
cv2.CascadeClassifier(
        #     'assets/lbpcascade_f
rontalface.xml')
        faces =
haar_classifier.detectMultiScal
e(
            image_gray,
scaleFactor=1.3,
minNeighbors=5, minSize=(30,
30),
flags=cv2.CASCADE_SCALE_I
MAGE)
        if len(faces) == 0:
            return None, None
        (x, y, w, h) = faces[0]
        if return_img_gray:
            return
image_gray[y:y+w, x:x+h]
        else:
            return image[y:y+w,
x:x+h]
    except:
        return None, None
  def predict_image(self, img):
    try:
        face =
self.face_detection(img)
        if face[0] is None:
            return {'detect':
None}
        label, confidence =
self.model.predict(face)
        print(label, confidence)
        if confidence <= 40:
            return {'detect': True,
'label': label}
        else:
            return {'detect':
False}
    except:
        return {'detect': None}
  def cv2_base64(self, image):
    base64_str =
cv2.imencode('.jpg',
image)[1].tostring()
    base64_str =
base64.b64encode(base64_str)
    return base64_str
  def base64_cv2(self,
base64_string):
    encoded_data =
base64_string.split(',')[1]
    nparr =
np.fromstring(base64.b64decod
e(str(encoded_data)), np.uint8)
    img =
cv2.imdecode(nparr,
cv2.IMREAD_ANYCOLOR)
    return img
```

M. *Main.py*

```
import os
from fastapi.middleware.cors
import CORSMiddleware
from config.db import engine,
Base
from routes.index import
authRoute, userRoute,
kandidatRoute, pemilihRoute,
daftarVoteRoute, voteRoute,
totalRoute, cetakRoute
from app import app
import routes.websocket
from dotenv import load_dotenv
load_dotenv()
Base.metadata.create_all(bind
=engine)
app.add_middleware(
  CORSMiddleware,
  allow_origins=["http://local
host",
          "http://localhost:80
80",
os.getenv('CLIENT_URL')],
  allow_credentials=True,
  allow_methods=["*"],
  allow_headers=["*"],
)
@app.get('/', tags=['Index'])
async def index():
  return "E-Voting REST API"
app.include_router(authRoute)
```

```
app.include_router(userRoute)
app.include_router(kandidatRo
ute)
app.include_router(pemilihRout
e)
app.include_router(daftarVoteR
oute)
app.include_router(voteRoute)
app.include_router(totalRoute)
app.include_router(cetakRoute)
```

2. Desain

A. Indexs.js

```
import Vue from "vue";
import VueRouter from "vue-
router";
import store from "@/store";
Vue.use(VueRouter);
const isLogin = (to, from, next) =>
{
  const { login, role } =
store.state.authModule;
  try {
    if (to.name == "Login") {
      if (login) next(role);
    } else {
      if (!login) next("/");
      if (to.name.toLowerCase() !=
role) {
        next(role);
      }
    }
  } catch (error) {
    // console.log(error);c
  }
  next();
};
const routes = [
  {
    path: "/",
    name: "Login",
    beforeEnter: isLogin,
    component: () =>
import("../views/Login.vue"),
  },
  {
    path: "/logout",
    beforeEnter: (to, from, next) =>
{
      store.dispatch("authModule/log
out");
      next("/");
    },
  },
  {
    path: "/admin",
    component: () =>
import("../views/admin/index.vue"),
    beforeEnter: isLogin,
    children: [
      {
        path: "",
        name: "admin",
        component: () =>
import("../views/admin/dashboard.v
ue"),
      },
      {
        path: "kandidat/",
        component: () =>
import("../views/admin/kandidat.vu
e"),
      },
      {
        path: "pemilih/",
        component: () =>
import("../views/admin/pemilih.vue
"),
      },
      {
        path: "daftarvote/",
        component: () =>
import("../views/admin/daftarVote.v
ue"),
      },
      {
        path: "kotaksuara/",
        component: () =>
import("../views/admin/kotakSuara.
vue"),
      },
      {
```

```
      path: "hitungcepat/",
      component: () =>
import("../views/admin/hitungCepat
.vue"),
    },
    {
      path: "laporan/",
      component: () =>
import("../views/admin/laporan.vue
"),
    },
    {
      path: "user/",
      component: () =>
import("../views/admin/user.vue"),
    },
  ],
 },
 {
  path: "/petugas",
  component: () =>
import("../views/petugas/index.vue"
),
  beforeEnter: isLogin,
  children: [
   {
    path: "",
    name: "petugas",
    component: () =>
import("../views/admin/dashboard.v
ue"),
   },
   {
    path: "kandidat/",
    component: () =>
import("../views/admin/kandidat.vu
e"),
   },
   {
    path: "pemilih/",
    component: () =>
import("../views/admin/pemilih.vue
"),
   },
   {
    path: "daftarvote/",
    component: () =>
import("../views/admin/daftarVote.v
ue"),
   },
   {
    path: "kotaksuara/",
    component: () =>
import("../views/admin/kotakSuara.
vue"),
   },
   {
    path: "hitungcepat/",
    component: () =>
import("../views/admin/hitungCepat
.vue"),
   },
   {
    path: "laporan/",
    component: () =>
import("../views/admin/laporan.vue
"),
   },
  ],
 },
 {
  path: "/pemilih",
  component: () =>
import("../views/pemilih/index.vue"
),
  beforeEnter: isLogin,
  children: [
   {
    path: "",
    name: "pemilih",
    meta: { text: "Home" },
    component: () =>
import("../views/pemilih/home.vue"
),
   },
   {
    path: "vote/",
    name: "pemilih-vote",
    meta: { text: "Daftar Vote" },
    component: () =>
import("../views/pemilih/vote.vue"),
   },
```

```
    {
      path: "akun/",
      name: "akun",
      meta: { text: "Akun" },
      component: () =>
import("../views/pemilih/akun.vue")
,
    },
    {
      path: "facerecognition/",
      name: "face-recognition",
      meta: { text: "Daftar Face
Recognition" },
      component: () =>
import("../views/pemilih/faceRecog
nition.vue"),
    },
  ],
 },
];
const router = new VueRouter({
 // mode: "history",
 base: process.env.BASE_URL,
 routes,
});
export default router;
```

B. Login.js

```
    <template>
      <v-container fluid fill-height
    class="accent">
        <v-card class="mx-auto"
    width="90%">
          <v-card-title class="text-
    center justify-center py-6">
            <v-row>
              <v-col cols="12">
                <v-img

    src="@/assets/yukvoting.png"
                  class="mx-auto"
                  width="300px"
                ></v-img>
              </v-col>
            </v-row>
          </v-card-title>

    <v-tabs v-model="tab"
background-
color="transparent"
color="basil" grow>
      <v-tab v-for="item in
items" :key="item">
       {{ item }}
      </v-tab>
    </v-tabs>

    <v-tabs-items v-
model="tab">
      <!-- <v-tab-item v-
for="item in items"
:key="item">
       <v-card color="basil"
flat>
         <v-card-text>{{ text
}}</v-card-text>
       </v-card>
      </v-tab-item> -->
      <v-tab-item>
       <v-card
        elevation="0"
        :loading="loading"
        class="mx-auto mb-12"
        width="100%"
        max-width="500"
        v-bind:key="show"
       >
        <login-form
         v-if="show == 'login'"

@setLoading="setLoading"
          @setShow="setShow"
        >
        </login-form>

        <camera-form
         v-if="show ==
'camera'"

@setLoading="setLoading"
          @setShow="setShow"
        />
```

```
          <!-- <daftar-form
            v-if="show ==
'daftar'"

@setLoading="setLoading"
            @setShow="setShow"
          ></daftar-form> -->
        </v-card>
      </v-tab-item>

      <v-tab-item>
        <ChartSuara />
      </v-tab-item>
    </v-tabs-items>
  </v-card>

  <v-row>
    <v-col>
      <h4 class="py-2 white--
text text-center">
        &copy; {{ new
Date().getFullYear() }} —
        <strong>Aplikasi E-
Voting</strong>
      </h4>
    </v-col>
  </v-row>
 </v-container>
</template>

<script>
import LoginForm from
"../components/LoginForm.vue"
;
import CameraForm from
"../components/CameraForm.vu
e";
import ChartSuara from
"../components/ChartSuara.vue
";

export default {
 components: {
  LoginForm,
  CameraForm,
  ChartSuara,
 },
 data: () => ({
  show: "login",
  valid: true,
  loading: false,
  username: "",
  password: "",
  tab: null,
  items: ["Login", "Grafik
Perolehan Suara"],
 }),
 methods: {
  setLoading(loading) {
   this.loading = loading;
  },
  setShow(show) {
   this.show = show;
  },
 },
};
</script>
```

C. Admin

1. DaftarVote.js

```
<template>
 <v-container>
  <v-row>
   <v-col cols="12">
    <Table
     @tambah="tambah"
     @edit="edit"

    @hapus="showDialogAksi
    "
     :headers="headers"
     :items="items"
     itemKey="id"
     sortBy="id"
     :loading="loading"
     :showSelect="true"

    :selectedIndex.sync="select
    edIndex"

    :selectedFilter.sync="selec
    tedDaftarVote"
```

```
:isSelected.sync="isSelected"
      >
        <template v-slot:modal>
          <DialogForm
            :dialog="dialog"
            :loading="dialogLoading"
            :formTitle="formTitle"
            :formValid="valid"
            @closeDialog="closeDialog"
            @simpan="simpan"
          >
            <template v-slot:form>
              <!-- <v-form ref="form" v-model="valid" lazy-validation> -->
              <v-row>
                <v-col cols="12" sm="12" md="12">
                  <v-text-field
                    v-model="editedItem.nama"
                    label="Nama"
                    :rules="[(v) => !!v || 'Nama tidak boleh kosong']"
                    required
                  ></v-text-field>
                </v-col>
                <v-col cols="12" sm="12" md="12">
                  <v-textarea
                    v-model="editedItem.keterangan"
                    label="Keterangan"
                    :rules="[(v) => !!v || 'Keterangan tidak boleh kosong']"
                  ></v-textarea>
                </v-col>

                <v-col cols="6" sm="6" md="6">
                  <v-menu
                    ref="menuTanggalMulai"
                    v-model="menuTanggalMulai"
                    :close-on-content-click="false"
                    :return-value.sync="editedItem.tanggal_mulai"
                    transition="scale-transition"
                    offset-y
                    min-width="auto"
                  >
                    <template v-slot:activator="{ on, attrs }">
                      <v-text-field
                        v-model="editedItem.tanggal_mulai"
                        label="Tanggal Mulai"
                        prepend-icon="mdi-calendar"
                        readonly
                        v-bind="attrs"
```

```
                    v-on="on"
                    :rules="[
                      (v) => !!v
|| 'Tanggal mulai tidak
boleh kosong',
                    ]"
                    required
                  ></v-text-
field>
                </template>
                <v-date-
picker
                  v-
model="editedItem.tanggal
_mulai"
                  no-title
                  scrollable
                >
                  <v-
spacer></v-spacer>
                  <v-btn
                    text

color="primary"

@click="menuTanggalMul
ai = false"
                  >
                    Tutup
                  </v-btn>
                  <v-btn
                    text

color="primary"
                    @click="

$refs.menuTanggalMulai.s
ave(

editedItem.tanggal_mulai
                      )
                    "
                  >
                    OK
                  </v-btn>
                </v-date-
picker>
              </v-menu>
            </v-col>
            <v-col cols="6"
sm="6" md="6">
              <v-menu

ref="menuJamMulai"
                v-
model="menuJamMulai"
                :close-on-
content-click="false"
                :nudge-
right="40"
                :return-
value.sync="editedItem.ja
m_mulai"

transition="scale-
transition"
                offset-y
                max-
width="290px"
                min-
width="290px"
              >
                <template v-
slot:activator="{ on, attrs
}">
                  <v-text-field
                    v-
model="editedItem.jam_m
ulai"

label="Jam Mulai"
                    prepend-
icon="mdi-clock-time-four-
outline"
                    readonly
                    v-
bind="attrs"
                    v-on="on"
                    :rules="[
```

```
                    (v) => !!v
|| 'Jam mulai tidak boleh
kosong',
                  ]"
                  required
                ></v-text-
field>
              </template>
              <v-time-
picker
                v-
if="menuJamMulai"
                v-
model="editedItem.jam_m
ulai"
                full-width

@click:minute="

$refs.menuJamMulai.save(
editedItem.jam_mulai)
                "
              ></v-time-
picker>
            </v-menu>
          </v-col>

          <v-col cols="6"
sm="6" md="6">
            <v-menu

ref="menuTanggalSelesai"
              v-
model="menuTanggalSeles
ai"
              :close-on-
content-click="false"
              :return-
value.sync="editedItem.tan
ggal_selesai"

transition="scale-
transition"
              offset-y
              min-
width="auto"
              >
              <template v-
slot:activator="{ on, attrs
}">
                <v-text-field
                  v-
model="editedItem.tanggal
_selesai"

label="Tanggal Selesai"
                  prepend-
icon="mdi-calendar"
                  readonly
                  v-
bind="attrs"
                  v-on="on"
                  :rules="[
                    (v) => !!v
|| 'Tanggal selesai tidak
boleh kosong',
                  ]"
                  required
                ></v-text-
field>
              </template>
              <v-date-
picker
                v-
model="editedItem.tanggal
_selesai"
                no-title
                scrollable
              >
                <v-
spacer></v-spacer>
                <v-btn
                  text

color="primary"

@click="menuTanggalSele
sai = false"
                >
                  Tutup
                </v-btn>
                <v-btn
```

```
                    text
color="primary"
                    @click="
$refs.menuTanggalSelesai.save(
editedItem.tanggal_selesai
                    )
                    "
                  >
                    OK
                  </v-btn>
                </v-date-picker>
              </v-menu>
            </v-col>
            <v-col cols="6" sm="6" md="6">
              <v-menu
ref="menuJamSelesai"
                v-model="menuJamSelesai"
                :close-on-content-click="false"
                :nudge-right="40"
                :return-value.sync="editedItem.jam_selesai"
transition="scale-transition"
                offset-y
                max-width="290px"
                min-width="290px"
              >
                <template v-slot:activator="{ on, attrs }">
                  <v-text-field
                    v-model="editedItem.jam_selesai"
label="Jam Selesai"
                    prepend-icon="mdi-clock-time-four-outline"
                    readonly
                    v-bind="attrs"
                    v-on="on"
                    :rules="[
                      (v) => !!v || 'Jam selesai tidak boleh kosong',
                    ]"
                    required
                  ></v-text-field>
                </template>
                <v-time-picker
                  v-if="menuJamSelesai"
                  v-model="editedItem.jam_selesai"
                  full-width
@click:minute="
$refs.menuJamSelesai.save(editedItem.jam_selesai)
                  "
                ></v-time-picker>
              </v-menu>
            </v-col>
          </v-row>
          <!-- </v-form> -->
        </template>
      </DialogForm>
    </template>
  </Table>
```

```
    </v-col>
    <v-col cols="12"
sm="6" md="6">
      <Table

@tambah="dialogKandida
t = true"

@simpan="simpanKandid
at"
        @up="up"
        @down="down"

:headers="headersKandida
t"

:items.sync="itemListKand
idat"
        itemKey="id"
        sortBy="nomor"

:loading="loadingKandida
t"
        :showSearch="false"

:showCustomAksi="true"
        :hideFooter="true"

:isSelected="isSelected"
      >
        <template v-
slot:modal>
          <DialogCustom
           jenis="Kandidat"

@simpan="tutupModalKa
ndidat"

:dialogModel.sync="dialog
Kandidat"

:dialogLoading="dialogLo
ading"

:selected.sync="selectedKa
ndidat"

:items="itemsKandidat"
          />
        </template>
      </Table>
    </v-col>
    <v-col cols="12"
sm="6" md="6">
      <Table

@tambah="dialogPemilih
= true"

@simpan="simpanPemilih
"

:headers="headersPemilih
"

:items.sync="itemListPemil
ih"
        itemKey="id"
        sortBy="nama"

:loading="loadingPemilih"
        :showSearch="false"

:showCustomAksi="true"
        :hideFooter="true"

:isSelected="isSelected"
      >
        <template v-
slot:modal>
          <DialogCustom
           jenis="Pemilih"

@simpan="tutupModalPe
milih"

:dialogModel.sync="dialog
Pemilih"

:dialogLoading="dialogLo
ading"
```

```
:selected.sync="selectedPe
milih"

:items="itemsPemilih"
          />
        </template>
      </Table>
    </v-col>
  </v-row>
  <DialogAksi
    :dialog="dialogDelete"

:dialogLoading="dialogLo
ading"
    title="Anda yakin untuk
menghapus data ini?"
    @submit="hapus"

@closeDialog="closeDialo
g"
  />

  <v-snackbar
    :timeout="2000"
    v-
model="response.show"
    centered
    text
    outlined
    :color="response.type"
  >
    <v-alert
:type="response.type" text
shaped prominent>
      <strong v-
text="response.text"></str
ong>
    </v-alert>
  </v-snackbar>
 </v-container>
</template>

<script>
import Table from
"@/components/Table.vue"
;
import DialogForm from
"@/components/DialogFor
m.vue";
import DialogCustom from
"@/components/DialogCus
tom.vue";
import DialogAksi from
"@/components/DialogAksi
.vue";
import { mapState,
mapActions } from "vuex";

import DaftarVoteModel
from
"@/models/daftarVote";

export default {
 components: {
  Table,
  DialogForm,
  DialogCustom,
  DialogAksi,
 },
 data() {
  return {
   loading: true,
   loadingKandidat: true,
   loadingPemilih: true,
   headers: [
    {
     text: "#",
     align: "start",
     sortable: false,
     value: "index",
    },
    { text: "Nama Vote",
value: "nama" },
    { text: "Keterangan",
value: "keterangan" },
    { text: "Tanggal",
value: "tanggal" },
    { text: "Status", value:
"statusVote" },
```

```
      { text: "Aksi", value:
"aksi", sortable: false },
    ],
    headersKandidat: [
      { text: "Nomor Urut",
value: "nomor" },
      { text: "Foto
Kandidat", value:
"fotoKandidat" },
      { text: "Nama
Kandidat", value: "nama"
},
      { text: "Aksi", value:
"aksiKandidat" },
    ],
    headersPemilih: [
      {
        text: "#",
        align: "start",
        sortable: false,
        value: "index",
      },
      { text: "Nama
Pemilih", value: "nama" },
      { text: "Status", value:
"vote_nomor" },
    ],
    showPassword: false,
    itemListKandidat: [],
    itemListPemilih: [],
    menuTanggalMulai:
false,
    menuTanggalSelesai:
false,
    menuJamMulai: false,
    menuJamSelesai: false,
    selectedIndex: null,
    selectedDaftarVote:
null,
    isSelected: false,
    dialogKandidat: false,
    dialogPemilih: false,
    selectedKandidat: [],
    selectedPemilih: [],
  };
 },
 async created() {
   // cek jika data state
telah diload
   let requestPromise = [];

   if (!this.items.length)
requestPromise.push(this.g
etAll());

   if
(!this.itemsKandidat.length
)

requestPromise.push(this.$
store.dispatch("kandidatMo
dule/getAll"));

   if
(!this.itemsPemilih.length)

requestPromise.push(this.$
store.dispatch("pemilihMod
ule/getAll"));

   await
Promise.all(requestPromis
e);

   await
this.$store.commit("crudM
odule/initCrud", {
     model:
DaftarVoteModel,
     moduleName:
"daftarVoteModule",
   });

   this.loading = false;
   this.loadingKandidat =
false;
   this.loadingPemilih =
false;
 },
 computed: {
```

```
...mapState("daftarVoteModule", ["items"]),

...mapState("kandidatModule", { itemsKandidat: "items" }),
    //
...mapState("pemilihModule", { itemsPemilih: "items" }),

...mapState("crudModule", [
      "valid",
      "dialog",
      "dialogLoading",
      "dialogDelete",
      "editedIndex",
      "editedItem",
      "response",
    ]),
    formTitle() {
      return this.editedIndex === -1
        ? "Tambah Data Daftar Vote"
        : "Edit Data Daftar Vote";
    },
    itemsPemilih() {
      return this.$store.getters["pemilihModule/pemilihVerif"];
    },
  },
  watch: {
    selectedDaftarVote(val) {
      const item = this.items.find((item) => item.id == val);
      if (item) {
        this.itemListKandidat =
item.list_kandidat.map((kandidat) => {
          return {
            ...kandidat,

...this.itemsKandidat.find((k) => k.id == kandidat.id),
          };
        });

        this.itemListPemilih =
item.list_pemilih.map((pemilih) => {
          return {
            ...pemilih,

...this.itemsPemilih.find((k) => k.id == pemilih.id),
          };
        });
      } else {
        this.itemListKandidat = [];
        this.itemListPemilih = [];
      }
    },
    itemListKandidat(val) {
      const listIDKandidat = val.map((k) => k.id);

      const selected = this.itemsKandidat.reduce((prev, curr, i) => {
        if (listIDKandidat.includes(curr.id)) prev.push(i);

        return prev;
      }, []);

      this.selectedKandidat = selected;
    },
    itemListPemilih(val) {
```

```
      const listIDPemilih =
val.map((k) => k.id);

      const selected =
this.itemsPemilih.reduce((p
rev, curr, i) => {
        if
(listIDPemilih.includes(cur
r.id)) prev.push(i);

        return prev;
      }, []);

      this.selectedPemilih =
selected;
    },
  },
  methods: {

...mapActions("daftarVote
Module", ["getAll",
"simpanDataList"]),

...mapActions("crudModule
", [
      "tambah",
      "edit",
      "hapus",
      "simpan",
      "showDialogAksi",
      "closeDialog",
    ]),
    tutupModalKandidat() {
      this.itemListKandidat =
this.selectedKandidat.map(
(v, n) => {
        return {
...this.itemsKandidat.find((
_, i) => i == v), nomor: n
+ 1 };
      });

      this.dialogKandidat =
false;
    },
    tutupModalPemilih() {
      this.itemListPemilih =
this.selectedPemilih.map((v
) => {
        return {
...this.itemsPemilih.find((_,
i) => i == v), vote_nomor:
-1 };
      });

      this.dialogPemilih =
false;
    },
    async simpanKandidat()
{
      this.loadingKandidat =
true;

      const res = await
this.simpanDataList({
        id:
this.selectedDaftarVote,
        index:
this.selectedIndex,
        items:
this.itemListKandidat,
        jenis: "kandidat",
      });

      this.loadingKandidat =
false;

      this.cekResponse(res);
    },
    async simpanPemilih() {
      this.loadingPemilih =
true;

      const res = await
this.simpanDataList({
        id:
this.selectedDaftarVote,
        index:
this.selectedIndex,
        items:
this.itemListPemilih,
        jenis: "pemilih",
```

```
      });

      this.loadingPemilih =
false;
      this.cekResponse(res);
    },
    cekResponse(res) {
      let text =
res.data.message;
      let type = "success";
      if (res.status != 200
&& res.status != 201 &&
res.status != 202) {
        text = res.data.detail;
        type = "error";
      }


this.$store.commit("crudM
odule/setResponse", {
        show: true,
        text,
        type,
      });
    },
    up(index) {
      //fungsi untuk
menaikkan nomor urut
kandidat

this.itemListKandidat[index
].nomor--;

this.itemListKandidat[index
- 1].nomor++;


this.swap(this.itemListKand
idat, index, index - 1);
    },
    down(index) {
      //fungsi untuk
menurunkan nomor urut
kandidat

this.itemListKandidat[index
].nomor++;

this.itemListKandidat[index
+ 1].nomor--;


this.swap(this.itemListKand
idat, index, index + 1);
    },
    swap(arr, from, to) {
      arr.splice(from, 1,
arr.splice(to, 1,
arr[from])[0]);
    },
  },
};
</script>
```

2. Dashboard

```
<template>
  <v-container>
    <v-row>
      <v-col cols="12">
        <v-row class="no-
gutters">
          <v-icon
color="grey"> mdi-
calendar </v-icon>
          <h4 class="blue-
grey--text text--lighten-1">
            {{ tanggal }}
          </h4>
        </v-row>
      </v-col>
      <v-col cols="12"
class="text-center">
        <v-img
          class="mx-auto"

src="@/assets/yukvoting.p
ng"
          transition="scale-
transition"
          width="150"
        ></v-img>
```

```
        <h1 class="primary--
text">Selamat Datang Di
Aplikasi E-Voting</h1>
        <h3 class="primary--
text">
          Pemilihan Ketua RW
Kelurahan Cappagalung
Kecamatan Bacukiki Barat
Kota
          Parepare
        </h3>
      </v-col>
      <v-col
        v-for="(item, i) in
items"
        :key="i"
        cols="12"
        sm="4"
        md="4"
        class="pb-2"
      >
        <CardCustom
          :loading="loading"
          :color="item.color"
          :icon="item.icon"
          :title="item.title"
          :total="item.total"
        />
      </v-col>

      <v-col cols="12">
<ChartSuara /> </v-col>
    </v-row>
  </v-container>
</template>

<script>
import axios from "axios";
import moment from
"moment";
import CardCustom from
"../../components/CardCust
om.vue";
import ChartSuara from
"../../components/ChartSua
ra.vue";

export default {
  components: {
    CardCustom,
    ChartSuara,
  },
  data: () => {
    return {
      items: [
        {
          title: "Total
Kandidat",
          total: 0,
          color: "secondary",
          icon: "mdi-account-
tie",
          // loading:
this.loading,
        },
        {
          title: "Total Pemilih",
          total: 0,
          color: "orange",
          icon: "mdi-account-
group",
          // loading:
this.loading,
        },
        {
          title: "Total Vote
Aktif",
          total: 0,
          color: "primary",
          icon: "mdi-vote",
          // loading:
this.loading,
        },
      ],
      kandidat: 0,
      pemilih: 0,
      vote_aktif: 0,

      loading: true,
    };
  },
  async created() {
```

```
    // let requestPromise =
[this.loadData()];

    // if
(!this.itemsDaftarVote.lengt
h)
requestPromise.push(this.g
etAll());

    // await
Promise.all(requestPromis
e);

    // this.selectedDaftarVote
= 8;

    await this.loadData();

    this.loading = false;
  },

  computed: {
    tanggal() {
      moment.locale("id");
      return
moment().format("DD
MMMM YYYY");
    },
  },
  methods: {
    async loadData() {
      const res = await
axios("total");

      const { kandidat,
pemilih, vote_aktif } =
res.data;

      this.items[0].total =
kandidat;
      this.items[1].total =
pemilih;
      this.items[2].total =
vote_aktif;
    },
  },
};
</script>
```

3. **HitungCepat**

```
<template>
  <v-container>
    <v-row>
      <v-col cols="12">
        <Table

:showBtnTambah="false"
          :headers="headers"

:items="itemsHitungCepat
"
          itemKey="id"
          sortBy="id"
          :loading="loading"
        >
          <template v-
slot:aksi>
            <v-autocomplete
              clearable
              dense
              outlined

placeholder="Pilih Daftar
Vote"
              prepend-inner-
icon="mdi-vote"
              color="primary"
              class="mb-n6"

:items="itemsDaftarVote"
              item-text="nama"
              item-value="id"
              v-
model="selectedDaftarVot
e"
            ></v-
autocomplete>

            <v-tooltip bottom>
              <template v-
slot:activator="{ on, attrs
}">
                <v-icon
```

```
            class="ml-4"
            v-bind="attrs"
            v-on="on"

color="primary"

:disabled="!selectedDaftar
Vote"

@click.stop="getDataKota
kSuara"
          >
            mdi-refresh-
circle
          </v-icon>
        </template>
        <span>Refresh
Data</span>
      </v-tooltip>
    </template>
  </Table>
 </v-col>
</v-row>
</v-container>
</template>

<script>
import Table from
"@/components/Table.vue"
;
import { mapState,
mapActions } from "vuex";

export default {
 components: {
  Table,
 },
 data() {
  return {
   loading: true,
   headers: [
    {
     text: "Nomor Urut",
     align: "start",
     sortable: false,
     value: "nomor",
    },
    { text: "Nama
Kandidat", value: "nama"
},
    { text: "Foto
Kandidat", value: "foto" },
    { text: "Keterangan",
value: "keterangan" },
    { text: "Jumlah
Suara", value: "jumlah" },
   ],
   itemsHitungCepat: [],
   selectedDaftarVote:
null,

selectedDaftarVoteIndex:
null,
  };
 },
 async created() {
  // cek jika data state
telah diload
  if
(!this.itemsDaftarVote.lengt
h) await this.getAll();

  this.loading = false;
 },
 watch: {
  selectedDaftarVote(val)
{
   if (!val) {
    this.itemsHitungCepat
= [];
    return;
   }


this.selectedDaftarVoteInde
x =
this.itemsDaftarVote.findIn
dex(
    (v) => v.id == val
   );
```

```
this.getDataKotakSuara();
    },
  },
  computed: {

...mapState("daftarVoteMo
dule", {
      itemsDaftarVote:
"items",
    }),
  },
  methods: {

...mapActions("daftarVote
Module", ["getAll",
"getDataList"]),
    async
getDataKotakSuara() {
      this.loading = true;

      const res = await
this.getDataList({
        id:
this.selectedDaftarVote,
        index:
this.selectedDaftarVoteInde
x,
        jenis: "hitungcepat",
      });

      this.itemsHitungCepat
= res.data;

      this.loading = false;
    },
  },
};
</script>
```

4. **Index**

```
<template>
  <v-app>
    <v-app-bar
color="primary" dark
app>
      <v-app-bar-nav-icon
@click.stop="drawer =
!drawer"></v-app-bar-
nav-icon>
      <!-- <v-toolbar-title>{{
items[selectedItem].text
}}</v-toolbar-title> -->
      <v-toolbar-title>{{
selectedItem }}</v-toolbar-
title>
      <v-spacer></v-
spacer>

      <v-btn icon
to="/logout">
        <v-icon>mdi-
logout</v-icon>
      </v-btn>
    </v-app-bar>

    <v-navigation-drawer v-
model="drawer"
color="primary" app
bottom right dark>
      <v-list>
        <v-list-item link>
          <v-list-item-content>
            <v-list-item-title v-
text="nama"> </v-list-
item-title>
          </v-list-item-
content>
        </v-list-item>
      </v-list>
      <v-divider></v-
divider>
      <v-list nav dense>
        <v-list-item-group
active-class="text--accent-
4">
          <template v-
for="(item, i) in items">
            <v-subheader
              v-if="item.header"
              :key="`h-${i}`"
```

```
          v-
text="item.header"
          ></v-subheader>

          <v-list-item
           v-else-
if="!item.children"
           :to="item.href"
           :key="`l-${i}`"
           link
           exact
          >
           <v-list-item-icon>
            <v-icon v-
text="item.icon"></v-
icon>
           </v-list-item-
icon>
           <v-list-item-title v-
text="item.text"></v-list-
item-title>
          </v-list-item>

          <v-list-group
           v-else

:value="item.active"
           :key="item.text"
           :prepend-
icon="item.icon"
           no-action
           color="white"
          >
           <template v-
slot:activator>
            <v-list-item-
content>
             <v-list-item-title
v-text="item.text"></v-list-
item-title>
            </v-list-item-
content>
           </template>

           <v-list-item
                v-for="subItem
in item.children"

:to="subItem.href"

:key="subItem.text"
               >
                <v-list-item-title
v-text="subItem.text"></v-
list-item-title>

                <v-list-item-
icon>
                 <v-icon v-
text="subItem.icon"></v-
icon>
                </v-list-item-
icon>
               </v-list-item>
              </v-list-group>
             </template>
            </v-list-item-group>
           </v-list>
          </v-navigation-drawer>

          <v-main>
           <v-container
class="px-4 py-4">
            <transition
name="fade">
             <router-
view></router-view>
            </transition>
           </v-container>
          </v-main>

          <Footer />
         </v-app>
        </template>

        <script>
        import Footer from
"@/components/Footer.vue
";
        import { mapState } from
"vuex";
```

```
export default {
  components: { Footer },
  data() {
    return {
      drawer: true,
      // selectedItem: 0,
      items: [
        { text: "Dashboard", href: "/admin", icon: "mdi-view-dashboard" },
        {
          text: "Master Data",
          icon: "mdi-gamepad-circle-outline",
          active: false,
          children: [
            {
              text: "Data Kandidat",
              href: "/admin/kandidat",
              icon: "mdi-account-tie",
            },
            {
              text: "Data Pemilih",
              href: "/admin/pemilih",
              icon: "mdi-account-group ",
            },
            {
              text: "Daftar Vote",
              href: "/admin/daftarvote",
              icon: "mdi-vote ",
            },
          ],
        },

        {
          text: "Kotak Suara",
          href: "/admin/kotaksuara",
          icon: "mdi-package-variant",
        },
        {
          text: "Hitung Cepat",
          href: "/admin/hitungcepat",
          icon: "mdi-counter",
        },
        {
          text: "Laporan",
          href: "/admin/laporan",
          icon: "mdi-chart-box",
        },
        { header: "Setting" },
        {
          text: "Users",
          href: "/admin/user",
          icon: "mdi-account-cog",
        },
        {
          text: "Logout",
          href: "/logout",
          icon: "mdi-logout-variant",
        },
      ],
    };
  },
  computed: {

...mapState("authModule", ["nama"]),
    selectedItem() {
      let selected = this.items.findIndex(
        (item) => item.href == this.$route.path
      );
```

```
    const nested =
this.items
      .map((item, index) =>
        item.children
          ? [
              index,

item.children.findIndex(
                (child) =>
child.href ==
this.$route.path
              ),
            ]
          : item.href ==
this.$route.path
      )
      .filter(Boolean)[0];

    const res =
      selected >= 0
        ?
this.items[selected].text
        :
this.items[nested[0]].childr
en[nested[1]].text;

    return res;
  },
 },
};
</script>

<style lang="scss"
scoped>
.fade-enter-active,
.fade-leave-active {
 transition-property:
opacity;
 transition-duration: 0.25s;
}

.fade-enter-active {
 transition-delay: 0.25s;
}

.fade-enter,
.fade-leave-active {
 opacity: 0;
}
</style>
```

5. Kotak Suara

```
<template>
 <v-container>
  <v-row>
   <v-col cols="12">
    <Table

:showBtnTambah="false"
      :headers="headers"

:items="itemsKotakSuara"
      itemKey="id"
      sortBy="id"
      :loading="loading"
     >
      <template v-
slot:aksi>
       <v-autocomplete
        clearable
        dense
        outlined

placeholder="Pilih Daftar
Vote"
        prepend-inner-
icon="mdi-vote"
        color="primary"
        class="mb-n6"

:items="itemsDaftarVote"
        item-text="nama"
        item-value="id"
        v-
model="selectedDaftarVot
e"
       ></v-
autocomplete>

       <v-tooltip bottom>
        <template v-
slot:activator="{ on, attrs
}">
```

```
            <v-icon
              class="ml-4"
              v-bind="attrs"
              v-on="on"

color="primary"

:disabled="!selectedDaftar
Vote"

@click.stop="getDataKota
kSuara"
            >
              mdi-refresh-
circle
            </v-icon>
          </template>
          <span>Refresh
Data</span>
        </v-tooltip>
      </template>
    </Table>
  </v-col>
 </v-row>
</v-container>
</template>

<script>
import Table from
"@/components/Table.vue"
;
import { mapState,
mapActions } from "vuex";

export default {
 components: {
  Table,
 },
 data() {
  return {
   loading: true,
   headers: [
    {
     text: "#",
     align: "start",
     sortable: false,
     value: "index",
    },
    { text: "Nama
Pemilih", value: "nama" },
    { text: "Username",
value: "username" },
    { text: "Waktu
Memilih", value: "waktu" },
   ],
   itemsKotakSuara: [],
   selectedDaftarVote:
null,

selectedDaftarVoteIndex:
null,
  };
 },
 async created() {
  // cek jika data state
telah diload
  if
(!this.itemsDaftarVote.lengt
h) await this.getAll();

  this.loading = false;
 },
 watch: {
  selectedDaftarVote(val)
{
   if (!val) {
    this.itemsKotakSuara
= [];
    return;
   }


this.selectedDaftarVoteInde
x =
this.itemsDaftarVote.findIn
dex(
    (v) => v.id == val
   );


this.getDataKotakSuara();
  },
```

```
  },
  computed: {

...mapState("daftarVoteMo
dule", {
      itemsDaftarVote:
"items",
    }),
  },
  methods: {

...mapActions("daftarVote
Module", ["getAll",
"getDataList"]),
    async
getDataKotakSuara() {
      this.loading = true;

      const res = await
this.getDataList({
        id:
this.selectedDaftarVote,
        index:
this.selectedDaftarVoteInde
x,
        jenis: "kotaksuara",
      });

      this.itemsKotakSuara =
res.data;

      this.loading = false;
    },
  },
};
</script>
```

6. FaceRecognition

```
<template>
  <v-container>
    <v-card elevation="1"
class="mx-auto mb-12"
width="100%" max-
width="700">
      <v-card-title>
        <v-alert
          :value="error"
          dense
          outlined
          border="left"
          type="error"
          elevation="2"
        >
          <v-row
align="center">
            <v-col
class="grow"> {{ alert }}
</v-col>
            <v-col
class="shrink">
              <v-btn v-
show="btnErr"
@click="startCamera">ak
tifkan</v-btn>
            </v-col>
          </v-row>
        </v-alert>

        <v-alert

:value="alertDetection.sho
w"

:type="alertDetection.type"
          transition="scale-
transition"
          dense
          outlined
          border="left"
          elevation="2"
        >
          <span v-
html="alertDetection.text"
></span>
        </v-alert>
        <v-progress-linear

:active="progress.active"

:value="progress.value"
        ></v-progress-
linear>
      </v-card-title>
```

```
      <v-card-text>
        <v-container>
          <v-row
ref="container" align-
content="center"
justify="center">
            <video
              v-
show="cameraReady"
              ref="video"
              id="video"
              autoplay="true"
            ></video>
            <canvas
style="display: none"
id="preview"></canvas>
          </v-row>
        </v-container>
      </v-card-text>
    </v-card>
  </v-container>
</template>

<script>
import axios from "axios";
export default {
  data() {
    const id =
this.$store.state.authModul
e.id;
    const BASE_URL =
axios.defaults.baseURL
      .replace("http", "ws")
      .replace("https",
"wss");
    return {
      cameraReady: false,
      error: false,
      btnErr: false,
      alert: "Silahkan
aktifkan kamera terlebih
dahulu!",
      alertDetection: {
        show: false,
        type: "warning",
        text: "Tidak ada wajah
yang terdeteksi",
      },
      progressDetection: 0,
      response: null,
      socket: new
WebSocket(`${BASE_URL}
regisFace/${id}`),
    };
  },
  async mounted() {
    navigator.getUserMedia
=

navigator.getUserMedia ||

navigator.webkitGetUserM
edia ||

navigator.mozGetUserMedi
a ||

navigator.msGetUserMedia
;

    this.startCamera();

    this.canvas =
document.getElementById(
"preview");
    this.context =
this.canvas.getContext("2d
");
  },
  beforeDestroy() {
    if (!this.error) this.stop();

clearInterval(this.detection
Interval);
    this.socket.close();
  },
  computed: {
    progress() {
      return {
```

```
        active:
this.alertDetection.type
=== "info",
        value:
this.progressDetection,
      };
    },
  },
  methods: {
    settingSize(width, height)
{
      this.canvas.width =
width;
      this.canvas.height =
height;

      this.context.width =
this.canvas.width;
      this.context.height =
this.canvas.height;
    },
    startCamera() {

navigator.mediaDevices
        .getUserMedia({
          video: true,
          audio: false,
        })
        .then((stream) => {
          try {

this.$refs.video.srcObject =
stream;
        } catch (error) {
          this.$refs.video.src
=
URL.createObjectURL(stre
am);
        }


console.log(stream.getVide
oTracks()[0].getSettings());

        const { width, height
} =
stream.getVideoTracks()[0]
.getSettings();


this.settingSize(width,
height);

        this.cameraReady =
true;

        console.log("Camera
connected");


this.socket.onmessage =
this.stream;

        this.stop = () => {

stream.getVideoTracks().fo
rEach((track) => {
            track.stop();
          });
        };

        this.detectionInterval
= setInterval(() => {

this.draw(this.$refs.video,
this.context);
        }, 500);
      })
      .catch((err) => {
        this.error = true;
        this.alert =
          "Kamera telah di
blok, silahkan aktifkan
kamera secara manual!";
        this.btnErr = true;
        console.log(err);
      });
    },
    draw(video, context) {

context.drawImage(video,
```

```
0, 0, context.width,
context.height);

this.socket.send(this.canvas
.toDataURL());
   },
   async stream(message) {
    console.log(message);
    const { data } =
message;

    const res =
JSON.parse(data);


this.alertDetection.show =
true;
    this.alertDetection.text
= res.message;
    this.alertDetection.type
= res.detect == null ?
"error" : "warning";

    if (res.detect == true) {
     this.progressDetection
= res.progress;

this.alertDetection.type =
"info";

      if
(this.progressDetection <
100) return;


clearInterval(this.detection
Interval);

      // this.socket.close();


this.setFaceRecognition();


this.alertDetection.type =
"success";
      this.alertDetection.text
= "Data wajah berhasil
didaftar";
    }
   },
   setFaceRecognition() {
    const data =
this.$store.state.pemilihMo
dule.data;

    this.$store.commit(

"pemilihModule/setData",
      Object.assign(data, {
face_recognition: true })
    );

    setTimeout(() => {
     this.$router.go(-1);
    }, 2000);
   },
  },
};
</script>

<style lang="scss"
scoped>
video {
 width: 100%;
 min-width: 300px;
 background-color:
rgb(131, 128, 128);
}
</style>
```

7. **Vote**

```
<template>
 <v-row align="center"
justify="center">
   <v-col cols="12"
class="text-center">
    <h1 v-
text="daftarVote.nama"><
/h1>
   </v-col>
   <v-col cols="12">
```

```
    <countdown :end-
time="waktuSelesai">
      <span
       class="font-weight-
bold"
       slot="process"
       slot-scope="{
timeObj }"
       >{{
        `Sisa Waktu:
${timeObj.d > 0 ?
timeObj.d + " Hari" : ""}
${
          timeObj.h
         } Jam ${timeObj.m}
Menit ${timeObj.s} Detik`
       }}</span
      >
      <span slot="finish"
class="font-weight-bold"
       >Voting Telah
Berakhir!</span
      >
    </countdown>
  </v-col>

  <v-col>
   <v-item-group active-
class="primary">
     <v-container>
      <v-row
justify="center">
        <v-col
         v-for="(kandidat,
i) in
daftarVote.list_kandidat"
         :key="kandidat.id"
         class="d-flex
child-flex"
         cols="12"
         sm="4"
         md="4"
        >
         <v-hover v-slot="{
hover }">
          <v-card
:elevation="hover ? 12 :
2">
           <v-img

:src="getImgKandidat[i]"
            aspect-
ratio="1"
            class="grey
lighten-2"
           >
            <v-btn
             absolute
             class="mt-
10"
             color="grey
darken-2"
             large
             fab
             left
             top
            >
             <div
class="text-h4 font-weight-
bold white--text">
              {{
kandidat.nomor }}
             </div>
            </v-btn>
            <template v-
slot:placeholder>
             <v-row
              class="fill-
height ma-0"

align="center"

justify="center"
             >
              <v-
progress-circular

indeterminate

color="grey lighten-5"
```

```
                    ></v-
progress-circular>
                  </v-row>
                </template>
              </v-img>
              <v-card-text
class="pt-6"
style="position: relative">
                <v-scroll-x-
reverse-transition>
                  <v-btn
                    v-
if="belumMemilih"
                    absolute

color="primary"

class="white--text"
                    fab
                    large
                    right
                    top
                    @click="

SelectedkandidatIndex = i;
                      dialog =
true;
                    "
                  >
                    <v-
icon>mdi-vote</v-icon>
                  </v-btn>
                </v-scroll-x-
reverse-transition>
                <v-scroll-x-
reverse-transition>
                  <v-btn
                    v-if="

telahMemilih &&

kandidat.nomor ==
daftarVote.vote_nomor
                    "
                    absolute

color="primary"

class="white--text"
                    small
                    right
                    top
                  >
                    <v-icon
left>mdi-check-
decagram</v-icon> Dipilih
                  </v-btn>
                </v-scroll-x-
reverse-transition>

                <template v-
if="!loading">
                  <h3

class="overline text-h6
font-weight-bold primary--
text mb-2"
                  >
                    {{
listDataKandidat[i].nama
}}
                  </h3>
                  <div
class="font-weight-light
mb-2">
                    {{
listDataKandidat[i].ketera
ngan }}
                  </div>
                </template>
                <v-skeleton-
loader
                  v-
if="loading"
                  class="mx-
auto"
                  max-
width="300"
                  type="list-
item-three-line"
```

```
                ></v-skeleton-
loader>
                </v-card-text>
              </v-card>
            </v-hover>
          </v-col>
        </v-row>
      </v-container>
    </v-item-group>
   </v-col>
   <DialogAksi
    :title="dialogTitle"
    :dialog="dialog"

:dialogLoading="dialogLo
ading"
    @closeDialog="dialog
= false"
    @submit="setVote"
   />

   <v-snackbar v-
model="response.show"
:multi-line="true" center
vertical>
     <strong v-
text="response.text"></str
ong>
   </v-snackbar>
  </v-row>
</template>

<script>
import { mapState } from
"vuex";

import DialogAksi from
"../../components/DialogAk
si.vue";

export default {
 components: { DialogAksi
},
 data() {
   return {
     listDataKandidat: [],
     SelectedkandidatIndex:
null,
     loading: true,
     dialog: false,
     dialogLoading: false,
     response: {
       show: false,
       text: null,
     },
   };
 },
 async created() {
   let requestPromise = [];

   if
(!this.itemsKandidat.length
)

requestPromise.push(this.$
store.dispatch("kandidatMo
dule/getAll"));


requestPromise.push(this.lo
adDataKandidat());

   await
Promise.all(requestPromis
e);

   this.loading = false;
 },
 computed: {

...mapState("daftarVoteMo
dule", ["items",
"selectedIndex"]),

...mapState("kandidatModu
le", { itemsKandidat:
"items" }),
   daftarVote() {
     return
this.items[this.selectedInde
x];
   },
```

```
    waktuSelesai() {
      return new Date(

`${this.daftarVote.tanggal_
selesai}
${this.daftarVote.jam_seles
ai}`
      );
    },
    dialogTitle() {
      if
(this.SelectedkandidatIndex
== null) return;
      const kandidat =

this.daftarVote.list_kandida
t[this.SelectedkandidatInde
x];

      return `Anda yakin
untuk memilih nomor urut
${kandidat.nomor}`;
    },
    belumMemilih() {
      return (
        !this.loading &&

!this.daftarVote.telah_memi
lih &&
        this.waktuSelesai >
new Date()
      );
    },
    telahMemilih() {
      return !this.loading &&
this.daftarVote.telah_memil
ih;
    },
    getImgKandidat() {
      return
this.daftarVote.list_kandida
t.map((kandidat) => {
        return (

`http://res.cloudinary.com/
${process.env.VUE_APP_
CLOUD_NAME}/image/up
load/` +

this.itemsKandidat.find((k)
=> k.id ==
kandidat.id).foto
        );
      });
    },
  },
  methods: {
    async
loadDataKandidat() {
      const kandidat =
this.daftarVote.list_kandida
t.map(async (daftar) => {
        const res = await
this.$store.dispatch(

"kandidatModule/getKandi
dat",
          daftar.id
        );

        return res.data;
      });

      this.listDataKandidat =
await
Promise.all(kandidat);
    },

    async setVote() {
      this.dialogLoading =
true;

      const voteNomor =

this.daftarVote.list_kandida
t[this.SelectedkandidatInde
x].nomor;

      const res = await
this.$store.dispatch("daftar
VoteModule/simpanVote", {
```

```
        idDaftarVote:
this.daftarVote.id,
        idPemilih:
this.$store.state.authModul
e.id,
        voteNomor,
      });

      if (res.status == 201) {
        this.response.text =
res.data.message;

this.$store.commit("daftarV
oteModule/setVoting", {
          index:
this.selectedIndex,
          voteNomor,
        });
      } else {
        this.response.text =
res.data.detail;
      }

      this.dialog = false;
      this.dialogLoading =
false;
      this.response.show =
true;
    },
  },
};
</script>
```

8. Laporan

```
<template>
  <v-container>
    <v-row>
      <v-col cols="12">
        <v-card
elevation="3">
          <v-card-title>
            <v-col cols="12"
sm="12" md="12">
              <v-autocomplete
                v-
model="selectedLaporan"
                item-
value="value"

:items="listLaporan"

placeholder="Pilih
Laporan"
                dense
                outlined
              >
              </v-
autocomplete>
            </v-col>

            <v-col
              cols="12"
              sm="12"
              md="12"
              class="d-flex flex-
row-reverse mt-n6"
            >
              <v-btn
                outlined
                color="primary"

:disabled="!cetak"

@click="cetakClick"
              >
                <v-icon left>
mdi-printer </v-icon>
Cetak
              </v-btn>
            </v-col>
          </v-card-title>
        </v-card>
      </v-col>
      <v-col> </v-col>
      <dialog-cetak

:laporan="selectedLapora
n"
        :dialog="dialog"

@closeDialog="dialog =
false"
```

```
    />
   </v-row>
  </v-container>
</template>

<script>
import axios from "axios";
import { Capacitor, Plugins
} from "@capacitor/core";

const { Browser } =
Plugins;

import DialogCetak from
"@/components/DialogCet
ak.vue";

export default {
  components: {
    DialogCetak,
  },
  data() {
    return {
      loading: true,
      selectedLaporan: null,
      cetak: false,
      dialog: false,
      listLaporan: [
        { text: "Daftar
Kandidat", value:
"kandidat" },
        { text: "Daftar
Pemilih", value: "pemilih"
},
        { text: "Daftar Vote",
value: "daftarvote" },
      ],
    };
  },
  watch: {
    selectedLaporan(val) {
      if (val == null) return;
      this.cetak = true;
    },
  },
  methods: {
    async cetakClick() {
      if
(Capacitor.getPlatform()
== "android") {
        const url =
`${axios.defaults.baseURL
|| ""}cetak/${
          this.selectedLaporan
        }`;
        await Browser.open({
url });
      } else {
        this.dialog = true;
      }
    },
  },
};
</script>
```