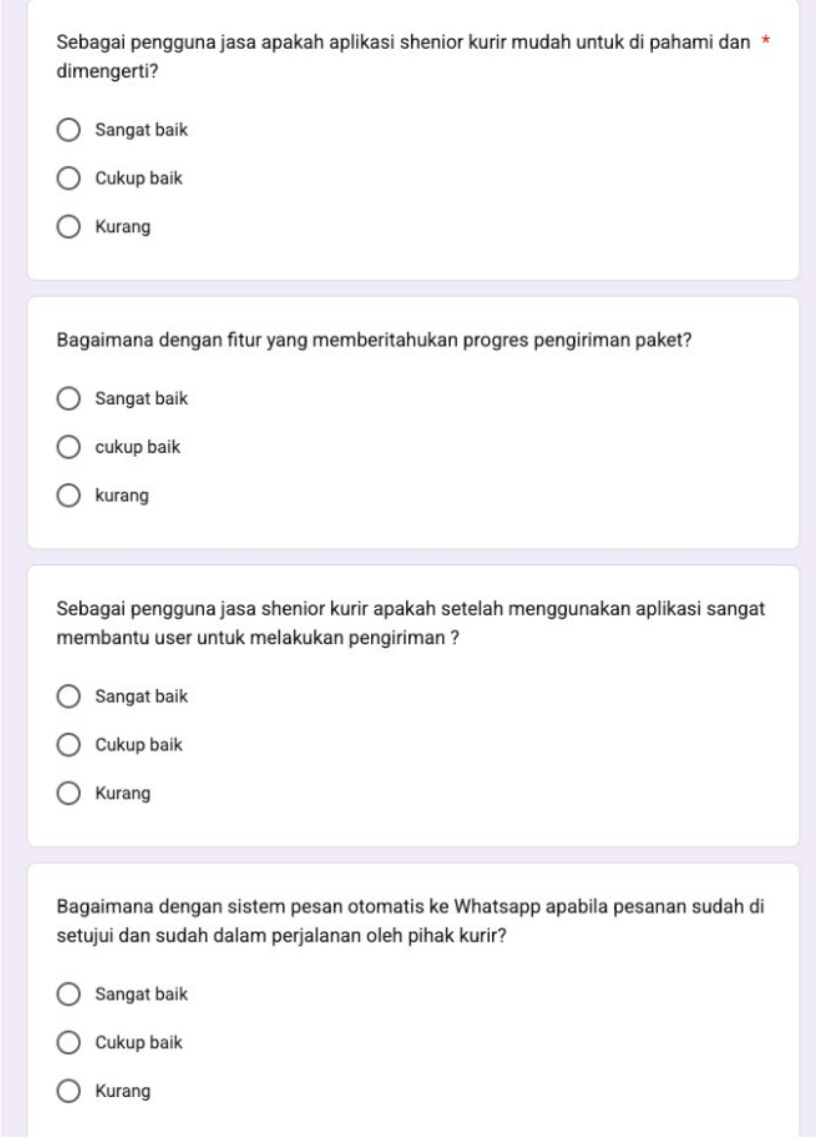
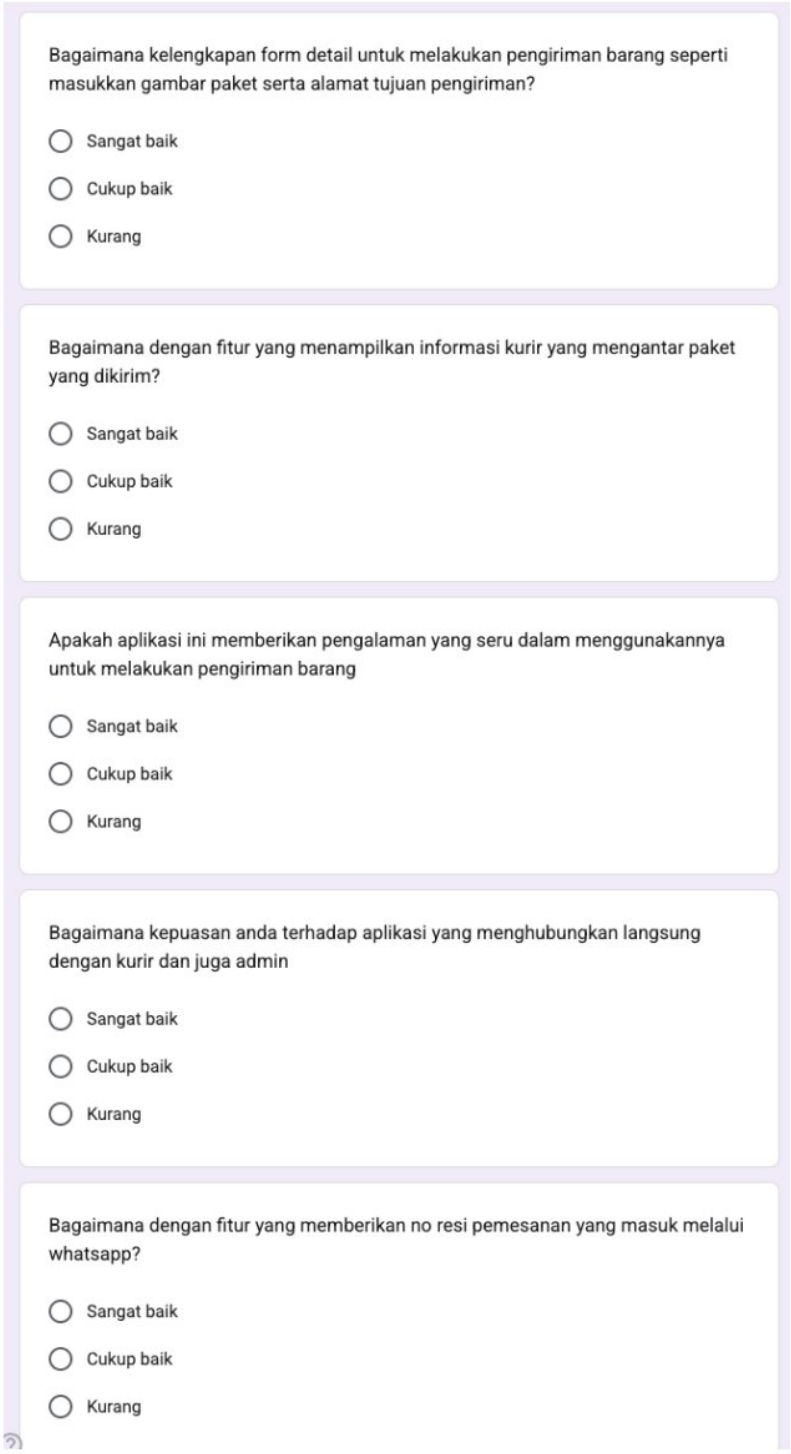
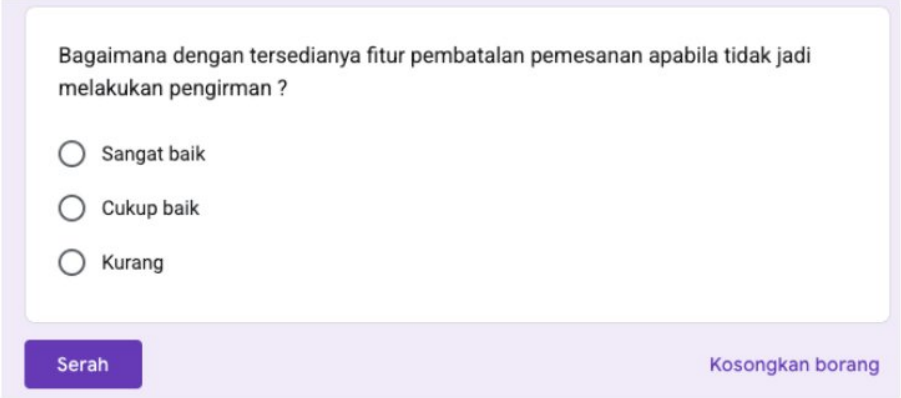
LAMPIRAN



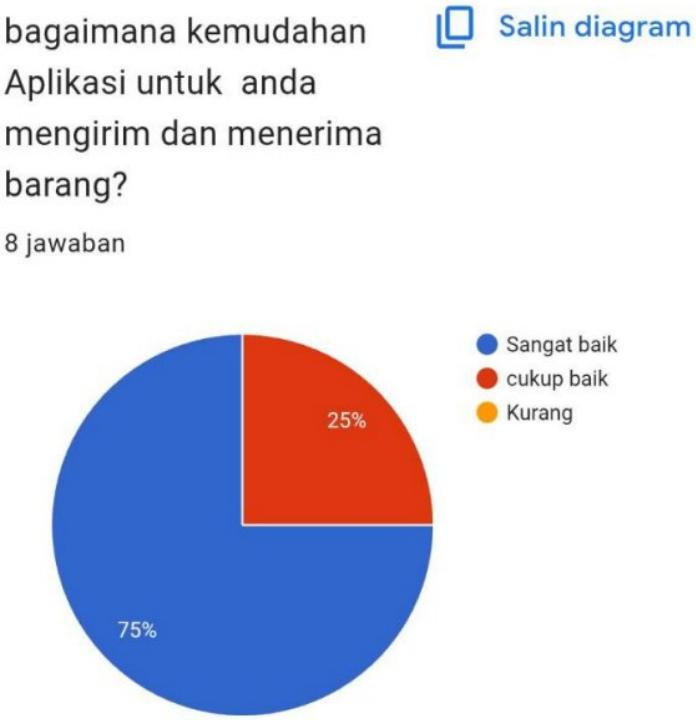
Gambar 6.1 Pertanyaan Kurir 1



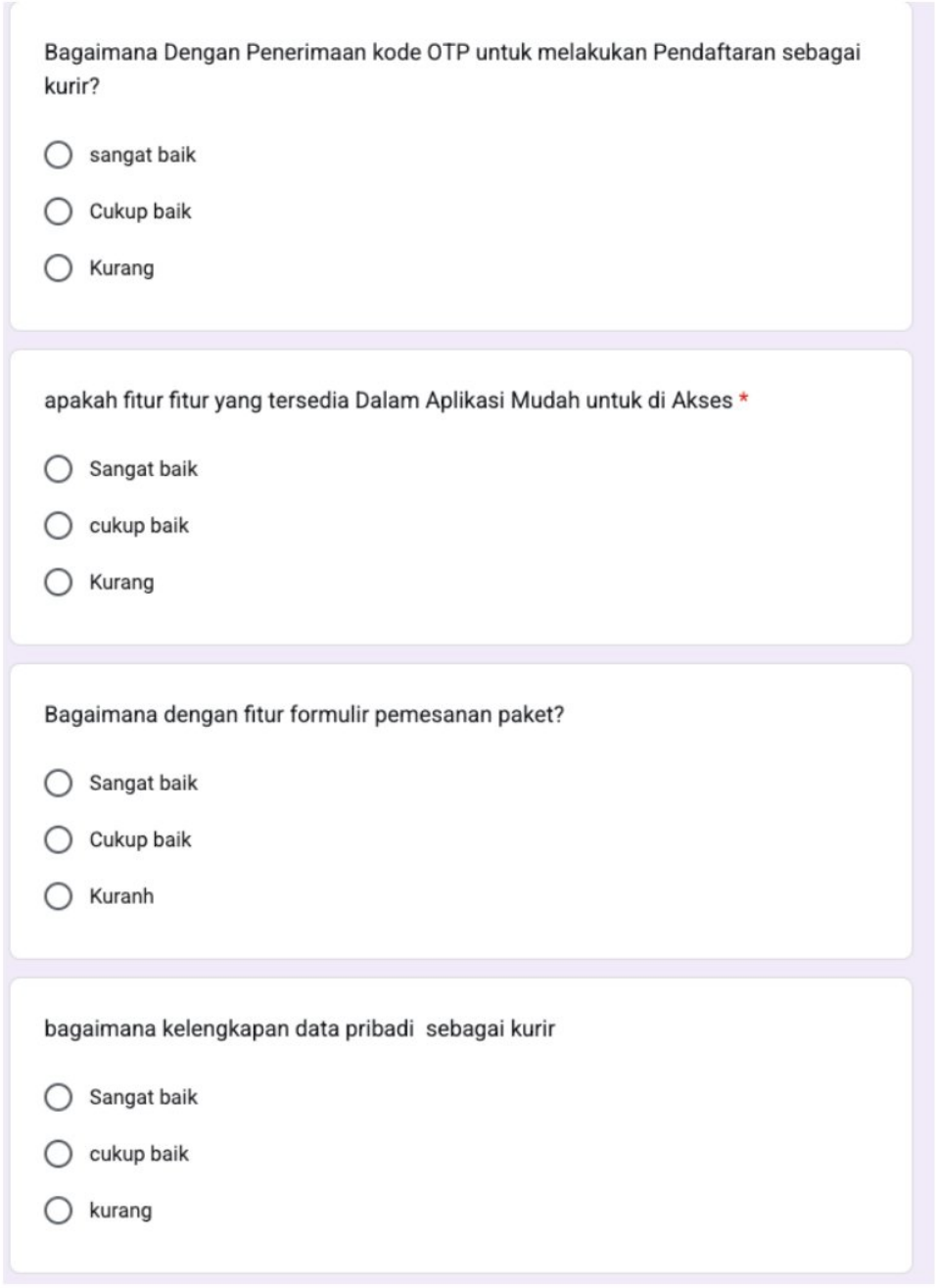
Gambar 6.2 Pertanyaan Kurir 2



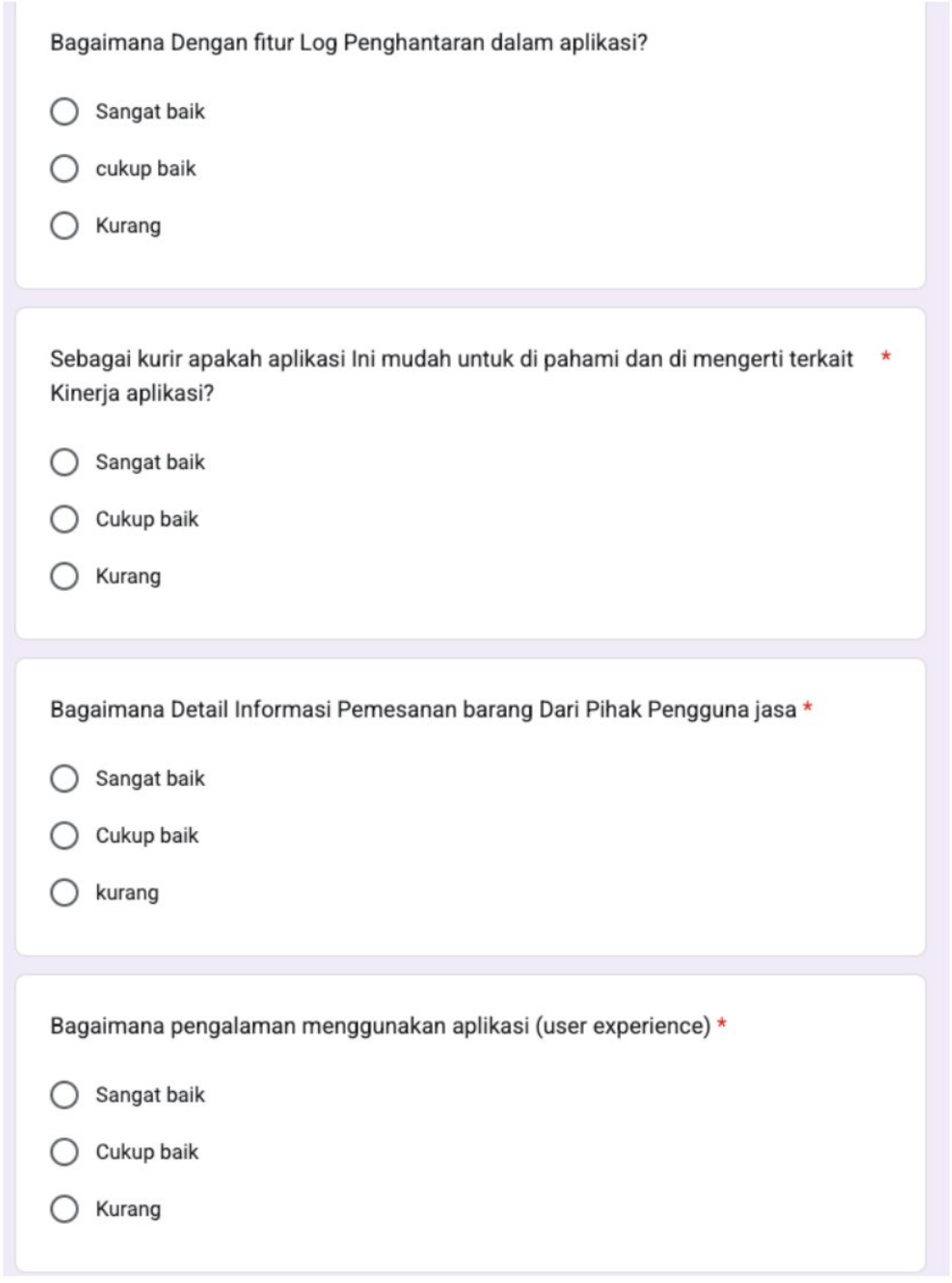
Gambar 6.3 Pertanyaan Kurir 3



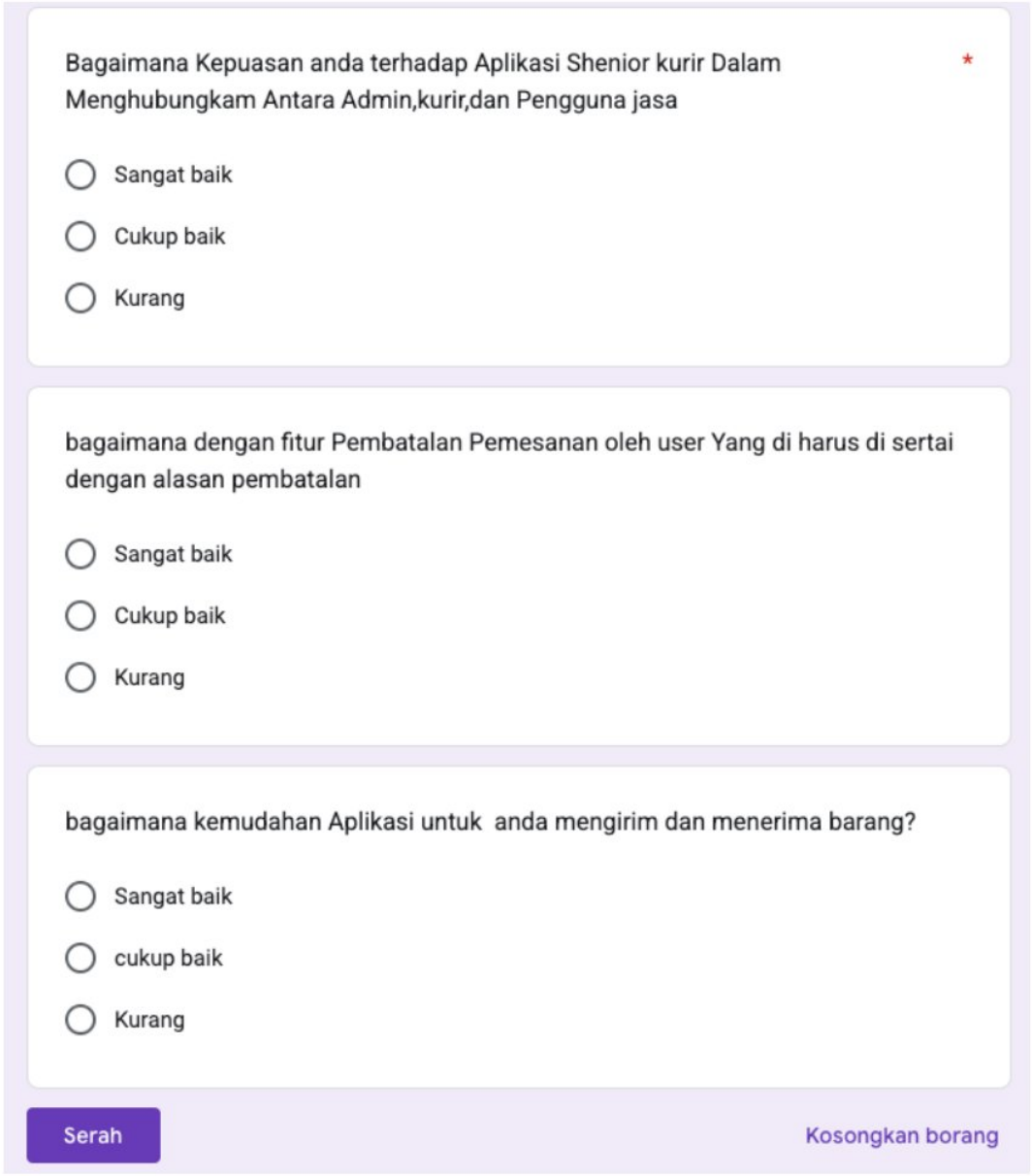
Gambar 6.4 Jawaban Kurir



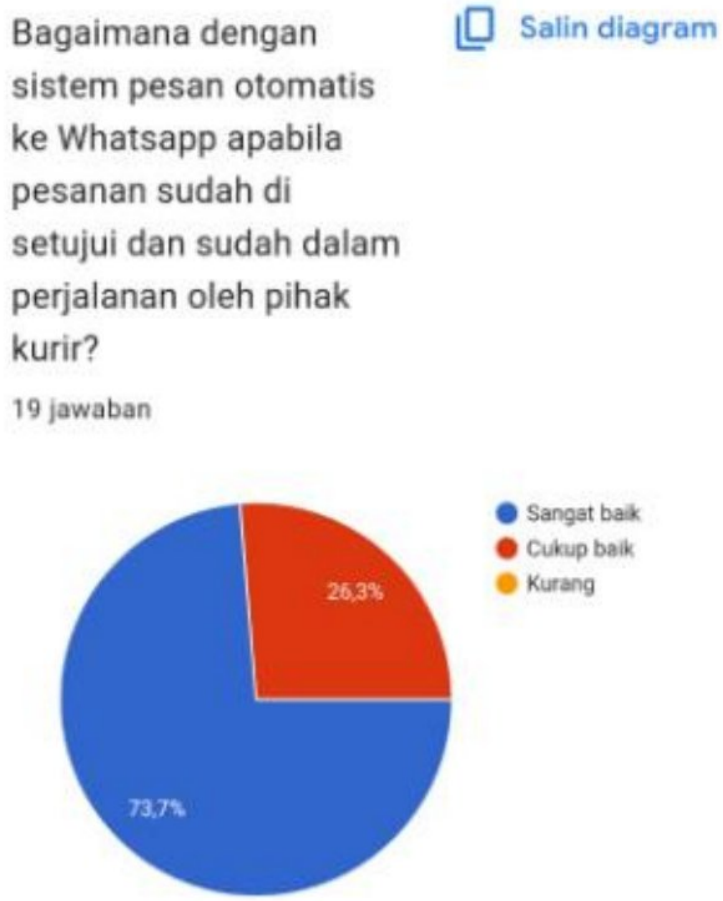
Gambar 6.5 Pertanyaan Pengguna 1



Gambar 6.6 Pertanyaan Pengguna 2



Gambar 6.7 Pertanyaan Pengguna 3



Gambar 6.8 Hasil Pertanyaan Pengguna

index.ts

```
// ini adalah file utama dari aplikasi ini
import express, { type Request, type
Response } from 'express';


// import cors from 'cors';
import fileUpload from 'express-fileupload';
import  {testDatabaseConnection} from
'./connection';
import path from 'path';

import { io, Socket } from 'socket.io-client';




const app = express();

import adminRouter from
'./routes/admin_router';
import userRouter from './routes/user_router';
import kurirRouter from './routes/kurir_router';


// this is for dotenv
import { config } from 'dotenv';
config();
console.log("diatas untuk dotenv");

const socket_client: Socket =
io(process.env.socket_server as string);



const port = process.env.PORT || 3011;




// Serve static files from the 'assets' directory
app.use(express.static(path.join(__dirname,
'assets')));

// Middleware for express-form-data
// app.use(formData.parse());
app.use(fileUpload({
   createParentPath: true, // Creates the
parent directory if it doesn't exist
}));
app.use(express.json());
app.use(express.urlencoded({ extended: true
}));

// app.options('*', cors())
// app.use(cors())



testDatabaseConnection();

// create an admin route
app.use('/admin', adminRouter);
app.use('/kurir', kurirRouter);
app.use('/', userRouter);

export { socket_client }

app.listen(port, async () => {
   console.log(`Server is running on port
${port}`);
});
```

1. admin_router.ts

```
// create admin router
import express from 'express';
import type { Request, Response } from
'express';
import type { UploadedFile } from 'express-
fileupload';
import KirimanModel from
'../models/kiriman_model';
import path from 'path';
import fs from 'fs';
import KurirModel from
'../models/kurir_model';
import UserModel from
'../models/user_model';
import axios from 'axios';
import { socket_client } from '../index';
```

```
const router = express.Router();

router.get('/', (req: Request, res: Response)
=> {
    res.sendFile(__dirname +
'/admin_ui/index.html');
});

router.get('/kurir', (req: Request, res:
Response) => {
    res.sendFile(__dirname +
'/admin_ui/kurir.html');
})

router.get('/kurir/data', (req: Request, res:
Response) => {
    KurirModel.find().then((data) => {
        res.json(data);
    });
})

router.get('/kurir/gambar/:no_telpon/:jenis',
(req: Request, res: Response) => {
    const { no_telpon, jenis } = req.params;
    if (!no_telpon) {
        res.status(400).json({ error: 'No Telpon
is required.' });
        return
    }
    KurirModel.findOne({ no_telpon
}).then((data) => {
        // if data not found
        if (!data) {
            res.status(404).json({ error: 'Kurir not
found.' });
            return
        }
        // Define file paths
        const motorGambar =
path.join(__dirname, '../images/' + jenis + '/' +
data.gambar_motor);
        const kurirGambar =
path.join(__dirname, '../images/kurir/' +
data.gambar_kurir);
        if (jenis == 'motor') {
            res.sendFile(motorGambar);
        } else {
            res.sendFile(kurirGambar);
        }
    })
})

router.get('/kurir/:no_telpon', (req: Request,
res: Response) => {
    const { no_telpon } = req.params;
    if (!no_telpon) {
        res.status(400).json({ error: 'No Telpon
is required.' });
        return
    }

    KurirModel.findOne({ no_telpon
}).then((data) => {
        // if data not found
        if (!data) {
            res.status(404).json({ error: 'Kurir not
found.' });
            return
        }
        res.json(data);

    });
})

router.post('/kurir', async (req: Request, res:
Response) => {
    try {
        const { dd_motor, nama, no_telpon,
jenis_kelamin } = req.body;
        const { gambar_kurir, gambar_motor } =
req.files as {
            gambar_kurir: UploadedFile;
            gambar_motor: UploadedFile;
        };

        // Check if kurir already exists by
no_telpon or dd_motor
        const existing = await
KurirModel.findOne({
            $or: [{ no_telpon }, { dd_motor }],
        });
```

```
        if (existing) {
            // await new Promise((resolve) =>
setTimeout(resolve, 2000));
            res.status(400).json({
                error: 'Nomor telepon atau motor
sudah terdaftar.',
            });
            return;
        }

        const random_5_char =
Math.random().toString(36).substring(2, 7);


        // Define file paths
        const kurirDir = path.join(__dirname,
'../images/kurir');
        const motorDir = path.join(__dirname,
'../images/motor');
        const kurirPath = path.join(kurirDir,
`${no_telpon}_kurir_${random_5_char}.jpg`);
        const motorPath = path.join(motorDir,
`${no_telpon}_motor_${random_5_char}.jpg`)
;

        // Ensure dirs exist
        fs.mkdirSync(kurirDir, { recursive: true
});
        fs.mkdirSync(motorDir, { recursive: true
});

        // Move files
        await gambar_kurir.mv(kurirPath);
        await gambar_motor.mv(motorPath);

        // Save to MongoDB
        const newKurir = new KurirModel({
            no_telpon,
            password: no_telpon,
            nama,
            jenis_kelamin,
            dd_motor,
            gambar_kurir:
`${no_telpon}_kurir_${random_5_char}.jpg`,
            gambar_motor:
`${no_telpon}_motor_${random_5_char}.jpg`,
        });

        await newKurir.save();
        // await 2 sec
        // await new Promise((resolve) =>
setTimeout(resolve, 2000));

        res.status(201).json({ message: 'Kurir
berhasil ditambahkan.' });

    } catch (err) {
        console.error(err);
        res.status(500).json({ error: 'Terjadi
kesalahan server.' });
    }
});

router.get('/check-kurir/tersedia', (req:
Request, res: Response) => {
    try {
        // search by status = "Tersedia" || null ||
undefined
        KurirModel.find({ status: { $in:
['Tersedia', null, undefined] } }).then((data) =>
{
            res.json(data);
        });
    } catch (error) {
        console.log(error);
        res.status(500).json([]);
    }
})

router.get('/paket-baru', async (req: Request,
res: Response) => {
    try {
        // search by status != "Dibatalkan Oleh
Admin"
        const response = await
KirimanModel.find({
            status: {
                $nin: ['Dibatalkan Oleh
Admin','Paket Telah Diterima
Penerima','Diterima Terverifikasi',
"Dibatalkan Oleh Pengirim"],
                $type: 'string'
            }
```

```
        });
        // console.log(response);
        res.status(200).json(response);
    } catch (error) {
        console.log(error);
        res.status(500).json([]);
    }
})

router.get('/paket-all', async (req: Request,
res: Response) => {
    try {
        // search by status != "Dibatalkan Oleh
Admin"
        const response = await
KirimanModel.find();
        // console.log(response);
        res.status(200).json(response);
    } catch (error) {
        console.log(error);
        res.status(500).json([]);
    }
})


router.delete('/batalkan-pengiriman/', async
(req: Request, res: Response) => {
    try {
        // const { id } = req.params;
        const { alasan, id } = req.body; //
optional reason from admin (e.g., from
SweetAlert input)
        // console.log(id, alasan);

        const updatedDoc = await
KirimanModel.findByIdAndUpdate(
            id,
            {
                $set: {
                    status: 'Dibatalkan Oleh Admin'
                },
                $push: {
                    timeline: {
                        status: 'Dibatalkan Oleh
Admin',
                        waktu: new Date(),
                        alasan: alasan || null
                    }
                }
            },
            { new: true } // Return the updated
document
        );
        // console.log(updatedDoc);
        const userData = await
UserModel.findById(updatedDoc?.id_pengiri
m);

        // console.log(userData)
        socket_client.emit('pembatalan_paket',
updatedDoc);

        try {
            await
axios.post('http://localhost:3012/send-
message', { number: userData?.no_telpon,
message: `🛵*_Kurir Shenior
🛵_*\nPengiriman anda dengan nomor resi
${updatedDoc?._id} telah dibatalkan oleh
admin\nAlasan: *${alasan || 'Belum ada
alasan'}* ❌❌` });
        } catch (err) {
            console.log(err);
        }


        res.status(200).json("response");
    } catch (error) {
        console.log(error);
        res.status(500).json([]);
    }
});

router.post('/tugaskan-kurir', async (req:
Request, res: Response) => {
    const { id_kiriman, id_kurir } = req.body;
    console.log(id_kiriman, id_kurir);
    try {
        const updatedDoc = await
KirimanModel.findByIdAndUpdate(
            id_kiriman,
            {
                $set: {
                    status: 'Kurir Telah Ditugaskan',
```

```
            id_kurir: id_kurir
          },
          $push: {
            timeline: {
              status: 'Kurir Telah
Ditugaskan',
              waktu: new Date(),
              // alasan: alasan || null
            }
          }
        },
        { new: true } // Return the updated
document
      );

      await KurirModel.findByIdAndUpdate(
        id_kurir,
        {
          $set: {
            status: 'Ditugaskan'
          }
        },
        { new: true } // Return the updated
document
      );

      console.log(updatedDoc);


socket_client.emit('tugaskan_kurir_server',
updatedDoc);

      try {
        const userData = await
UserModel.findById(updatedDoc?.id_pengiri
m);
        const kurirData = await
KurirModel.findById(id_kurir);
        await
axios.post('http://localhost:3012/send-
message', { number: userData?.no_telpon,
message: `🛵 * _Kurir Shenior
🛵_*\nPengiriman anda dengan nomor resi
${updatedDoc?._id} \nStatus : *Kurir Telah
DItugaskan*☑☑` });
        await
axios.post('http://localhost:3012/send-
message', { number: kurirData?.no_telpon,
message: `🛵 *_Kurir Shenior 🛵_*\nAnda
telah ditugaskan untuk mengirim
paket\nNomor Resi :
*${updatedDoc?._id}*\nAlamat Paket :
_*${updatedDoc?.alamat_pengirim}* _\nSila
buka situs Kurir Shenior dan login untuk
detail lebih lanjut ☑☑` });
      } catch (err) { console.log(err) }

      res.status(200).json(updatedDoc);

    } catch (err) {
      console.log(err);
      res.status(500).json([]);
    }
});


// ini untuk user
router.get('/user', (req: Request, res:
Response) => {
  res.sendFile(__dirname +
'/admin_ui/user.html');
});

router.get('/user/data', async (req: Request,
res: Response) => {
  try {
    const data = await UserModel.find();
    res.json(data);
  } catch (error) {
    console.log(error);
    res.status(500).json([]);
  }
});

router.get('/user/data/:id', async (req:
Request, res: Response) => {
  const { id } = req.params;
  console.log(id, "ini id");
  try {
    const data = await
UserModel.findById(id);
    res.json(data);
  } catch (error) {
    console.log(error);
```

```
        res.status(500).json([]);
    }
});

// ini untuk list penghantaran
router.get('/list', (req: Request, res:
Response) => {
    res.sendFile(__dirname +
'/admin_ui/list.html');
});


router.get('/login', (req: Request, res:
Response) => {
    res.sendFile(__dirname +
'/admin_ui/login.html');
});


export default router;
```

2. kurir_roueter.ts

```
import express from 'express';
import type { Request, Response } from
'express';
import PendafaranBaruModel from
'../models/pendaftaran_baru_model';
import KurirModel from
'../models/kurir_model';
import type { UploadedFile } from 'express-
fileupload';
import path from 'path';
import fs from 'fs';
import UserModel from
'../models/user_model';
import KirimanModel from
'../models/kiriman_model';
import axios from 'axios';
import { socket_client } from '../index';

const router = express.Router();

router.get('/', (req: Request, res: Response)
=> {
    res.sendFile(__dirname +
'/kurir_ui/index.html');
});

router.get('/penghantaran', (req: Request,
res: Response) => {
    res.sendFile(__dirname +
'/kurir_ui/penghantaran.html');
})

router.get('/login', (req: Request, res:
Response) => {
    res.sendFile(__dirname +
'/kurir_ui/login.html');
});

router.post('/login', async (req: Request, res:
Response) => {
    const { no_telpon, password } = req.body;

    try {
        const user = await KurirModel.findOne({
no_telpon, password });
        if (user) {
            res.status(200).json(user);
            return;
        }
        res.status(400).json('Nomor Telpon
atau Password Salah');
        return;
    } catch (error) {
        console.log("error di kurir login", error);
        res.status(500).json('Terjadi Kesalahan
Server');
        return;
    }

});

router.get('/:jenis/:gambar', (req: Request,
res: Response) => {
    const { jenis, gambar } = req.params;
    const imagePath = path.join(__dirname,
`../images/${jenis}/${gambar}`);

    if (fs.existsSync(imagePath)) {
        res.sendFile(imagePath);
    } else {
        res.status(404).send('Image not found');
```

```
    }

})

router.get('/gambar-paket/:id/:gambar', (req:
Request, res: Response) => {
    const { id, gambar } = req.params;
    const imagePath = path.join(__dirname,
`../images/kiriman/${id}/${gambar}`);

    if (fs.existsSync(imagePath)) {
        res.sendFile(imagePath);
    } else {
        res.status(404).send('Image not found');
    }

})

router.post('/check', async (req: Request,
res: Response) => {
    if (!req.body) {
        res.status(400).json('Bad Request');
        return;
    }
    const { _id, no_telpon, password,
createdAt } = req.body;

    try {
        // check the user by _id , no_telpon ,
password , createdAt
        const user = await KurirModel.findOne({
_id, no_telpon, password, createdAt });
        if (!user) {
            res.status(400).json('User not found');
            return;
        }

        res.status(200).json('Success');
        return;
    } catch (error) {
        console.log(error);
        res.status(500).json('Terjadi Kesalahan
Server');
        return
    }

})

router.get('/paket/:id/:status', async (req:
Request, res: Response) => {
    const { id, status } = req.params;
    try {
        const kirimanDoc = status == "ongoing"
? await KirimanModel.find({
            id_kurir: id,
            status: {
                $nin: ['Dibatalkan Oleh Admin',
'Paket Telah Diterima Penerima',"Diterima
Terverifikasi"],
                $type: 'string'
            }
        }) : await KirimanModel.find({ id_kurir: id
});
        // console.log(kirimanDoc);
        res.status(200).json(kirimanDoc);
    } catch (error) {
        console.log(error);
        res.status(500).json([]);
    }
})

router.get('/user', async (req: Request, res:
Response) => {
    const { id } = req.query;
    console.log(id , "ini di data user");
    // res.status(200).json(id);
    try {
        const user = await UserModel.findOne({
_id: id });
        res.status(200).json(user);
    } catch (error) {
        console.log(error);
        res.status(500).json();
    }

})

router.delete('/batalkan-pengiriman/:id_kurir',
async (req: Request, res: Response) => {
    const { id_kurir } = req.params;
    const { id, alasan } = req.body;
    // console.log(id_kurir , id , alasan);

    try {
```

```
        const updatedDoc = await
KirimanModel.findByIdAndUpdate(
            id,
            {
                $set: {
                    status: 'Dibatalkan Oleh Kurir ,
Admin mencari kurir baru',
                    id_kurir: null
                },
                $push: {
                    timeline: {
                        status: 'Dibatalkan Oleh Kurir
, Admin mencari kurir baru',
                        waktu: new Date(),
                        alasan: alasan || null,
                        id_kurir
                    }
                }
            },
            { new: true } // Return the updated
document

        );

        await KurirModel.findByIdAndUpdate(
            id_kurir,
            {
                $set: {
                    status: 'Tersedia'
                }
            },
            { new: true } // Return the updated
document
        )
        const userData = await
UserModel.findById(updatedDoc?.id_pengiri
m);
        // const kurirData = await
KurirModel.findById(id_kurir);


socket_client.emit('pembatalan_paket_kurir',
updatedDoc);

        try {
            await
axios.post('http://localhost:3012/send-
message', { number: userData?.no_telpon,
message: `🛵*_Kurir Shenior
🛵_*\nPengiriman anda dengan nomor resi
${updatedDoc?._id} telah dibatalkan oleh
kurir\nAdmin akan menugaskan kurir yang
baru\nHarap Bersabar'}* 🔄🔄` })
        } catch (error) {
            console.log(error)
        }

    } catch (error) {
        console.log(error)
        res.status(500).json();
    }


})


// kurir menyetujui pengambilan pengiriman
router.put('/mengambil-pengiriman/:id', async
(req: Request, res: Response) => {
    const { id: id_kiriman } = req.params;
    const { id_kurir } = req.body;


    // sini nnt berlaku update data
    // const kirimanData = await
KirimanModel.findById(id_kiriman);
    const kirimanDoc = await
KirimanModel.findByIdAndUpdate(
        id_kiriman,
        {
            $set: {
                id_kurir: id_kurir,
                status: 'Kurir Dalam Perjalanan',
            },
            $push: {
                timeline: {
                    status: 'Kurir Dalam Perjalanan',
                    waktu: new Date(),
                    alasan: null,
                    id_kurir
                }
            }
        },
```

```
      { new: true } // Return the updated
document
    );

    const userData = await
UserModel.findById(kirimanDoc?.id_pengirim
);
    const kurirData = await
KurirModel.findById(id_kurir);


    try {
      await
axios.post('http://localhost:3012/send-
message', { number: userData?.no_telpon,
message: `🛵*_Kurir Shenior_* 🛵
\nPengiriman anda dengan nomor resi
*${id_kiriman}* telash disetujui oleh kurir.\n
Kurir sekarang berangkat ke alamat pengirim
untuk mengambil paket, Silakan menunggu}
➡️➡️` })
      await
axios.post('http://localhost:3012/send-
message', { number: kurirData?.no_telpon,
message: `🛵*_Kurir Shenior_* 🛵 \nAnda
telah ditugaskan untuk mengambil paket
dengan nomor resi *${id_kiriman}* \nAlamat
Pengirim : *${kirimanDoc?.alamat_pengirim}*
\nNama Pengirim:
*${kirimanDoc?.nama_pengirim}*` })
    } catch (error) {
      console.log("Error Mengambil
Pengiriman | Kurir");
      console.log(error);
    }



    // notifikasi ke aplikasi (admin dan user)

socket_client.emit('kurir_mengambil_pengiri
man', kirimanDoc);


    res.status(200).json("ini mengambil
pengiriman");
})

router.post('/mengambil-pengiriman-dari-
pengirim-ke-penerima/:id', async (req:
Request, res: Response) => {
    const { id: id_kiriman } = req.params;
    const { id_kurir, status } = req.body;

    const { foto_paket } = req.files as {
      foto_paket: UploadedFile;

    };


    const random_5_char =
Math.random().toString(36).substring(2, 7);
    const nama_foto = `${random_5_char}-
${foto_paket.name}`;

    const paketDir = path.join(__dirname,
`../images/kiriman/${id_kiriman}`);
    const paketPath = path.join(paketDir,
`${nama_foto}`);

    fs.mkdirSync(paketDir, { recursive: true });

    // move photo
    await foto_paket.mv(paketPath);


    // const kirimanDoc = await
KirimanModel.findById(id_kiriman);
    const kirimanDoc = await
KirimanModel.findByIdAndUpdate(
      id_kiriman,
      {
        $set: {
          id_kurir: id_kurir,
          status: status == "mengirim" ?
'Mengirim Paket Ke Alamat Penerima' :
'Paket Telah Diterima Penerima',
        },
        $push: {
          timeline: {
            status: status == "mengirim" ?
'Mengirim Paket Ke Alamat Penerima' :
'Paket Telah Diterima Penerima',
```

```
                waktu: new Date(),
                alasan: null,
                id_kurir,
                gambar: nama_foto
            }
          }
        },
        { new: true } // Return the updated
document
    );


    const userData = await
UserModel.findById(kirimanDoc?.id_pengirim
);
    // const kurirData = await
KurirModel.findById(id_kurir);
    if (status == "menerima") {
        await KurirModel.findByIdAndUpdate(
            id_kurir,
            {
                $set: {
                    status: 'Tersedia'
                }
            }
        )
    }




    try {
        const mes = status == "mengirim" ?
`🛵 *_Kurir Shenior_* 🛵 \nPengiriman anda
dengan nomor resi *${id_kiriman}* telah
diambil oleh kurir.\nSilakan menunggu kurir
mengantarkan paket ke alamat penerima di
*${kirimanDoc?.alamat_penerima}*` :
`🛵 *_Kurir Shenior_* 🛵 \nPengiriman anda
dengan nomor resi *${id_kiriman}* telah
diterima oleh penerima.`
        await
axios.post('http://localhost:3012/send-
message', { number: userData?.no_telpon,
message: mes, img_stat: true, foto_paket:
paketPath, foto_name: foto_paket.name })
        // await
axios.post('http://localhost:3012/send-
message', { number: kurirData?.no_telpon,
message: `🛵 *_Kurir Shenior_* 🛵 \nAnda
telah mengambil paket dengan nomor resi
*${id_kiriman}* di alamat pengirim dan akan
menghantar ke \nAlamat Penerima :
*${kirimanDoc?.alamat_penerima}* \nNama
Penerima:
*${kirimanDoc?.nama_penerima}*`
,img_stat:true , foto_paket : paketPath,
foto_name : foto_paket.name})
    } catch (error) {
        console.log("Error Mengambil
Pengiriman di pengirim | Kurir");
        console.log(error);
    }



socket_client.emit('kurir_menghantar_ke_pen
erima', kirimanDoc);

    res.status(200).json("sukses");
})

// log penghantaran menu
router.get('/log', async (req: Request, res:
Response) => {
    res.sendFile(__dirname +
'/kurir_ui/log.html');
})

export default router;
```

3. user_router.ts

```
import express from 'express';
import type { Request, Response } from
'express';
import PendafaranBaruModel from
'../models/pendaftaran_baru_model';
import UserModel from
'../models/user_model';
import KirimanModel from
'../models/kiriman_model';
import KurirModel from
'../models/kurir_model';
```

```
import type { UploadedFile } from 'express-
fileupload';
import path from 'path';
import fs from 'fs';
import axios from 'axios';
import { socket_client } from '../index';

// import { io, Socket } from 'socket.io-client';


// const socket_client: Socket =
io("http://localhost:3014");

const router = express.Router();


router.get('/', (req: Request, res: Response)
=> {
   // socket_client.emit('scan_dia', 'test');
   res.sendFile(__dirname +
'/user_ui/index.html');
});



// router.get('/coba', async (req: Request, res:
Response) => {
//     res.status(200).json('success');
// })

router.get('/login', (req: Request, res:
Response) => {
   res.sendFile(__dirname +
'/user_ui/login.html');
})

router.post('/login', async (req: Request, res:
Response) => {
   const { no_telpon, password } = req.body;
   const user = await UserModel.findOne({
no_telpon, password });
   if (user) {
      console.log(user);
      res.status(200).json(user);
      return;
   }
   res.status(400).json('Nomor Telpon atau
Password Salah');
})

router.get('/daftar', (req: Request, res:
Response) => {
   res.sendFile(__dirname +
'/user_ui/daftar.html');
})

router.post('/daftar', async (req: Request, res:
Response) => {
   const { no_telpon, nama, password } =
req.body;
   // check the user
   const user = await UserModel.findOne({
no_telpon });
   if (user) {
      res.status(400).json('Nomor Telpon
sudah terdaftar');
      return;
   }

   // check the pendaftaran baru
   const pencarian_data = await
PendafaranBaruModel.findOne({ no_telpon
});
   if (pencarian_data) {
      // delete the pendaftaran baru with
no_telpon
      await
PendafaranBaruModel.deleteOne({
no_telpon });
   }

   const otp = Math.floor(Math.random() *
1000000);
   // create pendaftaran baru


   // create a 6 otp random otp code

   try {
      const response = await
axios.post('http://localhost:3012/send-otp', {
number: no_telpon, otp });
```

```
        const pendaftaran_baru = new
PendafaranBaruModel({ no_telpon, nama,
password, otp });
        await pendaftaran_baru.save();
        res.status(200).json('success');
        return;
    } catch (error: any) {
        // console.log(error);

res.status(error.response.status).json(error.r
esponse.data);
        return
    }



    // res.status(400).json('error dari
backend');
});

router.post('/verifikasi', async (req: Request,
res: Response) => {
    const { no_telpon, kode_otp } = req.body;
    const pencarian_data = await
PendafaranBaruModel.findOne({ no_telpon
});
    if (!pencarian_data) {
        res.status(400).json('Nomor Telpon
tidak ditemukan');
        return;
    }

    if (pencarian_data.otp == kode_otp) {
        const user = new UserModel({
no_telpon, nama: pencarian_data.nama,
password: pencarian_data.password });
        await user.save();
        await
PendafaranBaruModel.deleteOne({
no_telpon });
        res.status(200).json('success');
        return;
    }
    res.status(400).json('Kode OTP Salah');
    return
})

router.get('/user', (req: Request, res:
Response) => {
    res.sendFile(__dirname +
'/user_after_login_ui/index.html');
})

router.get('/user/gambar/:id/:gambar', (req:
Request, res: Response) => {
    const { id, gambar } = req.params;
    // console.log(id, gambar);
    const gambarPath = path.join(__dirname,
'../images/user/' + id + '/' + gambar);
    res.sendFile(gambarPath);
})

router.put('/user/:_id', async (req: Request,
res: Response) => {
    const { _id } = req.params;
    const { nama, alamat, ada_file } =
req.body;
    let gambar_name = "";
    if (ada_file > 0) {
        const { gambar } = req.files as { gambar:
UploadedFile };
        // console.log(gambar);
        const userDir = path.join(__dirname,
'../images/user/' + _id);
        if (!fs.existsSync(userDir)) {
            fs.mkdirSync(userDir);
        }
        // delete all file on userDir
        const files = fs.readdirSync(userDir);
        for (const file of files) {
            fs.unlinkSync(path.join(userDir, file));
        }

        const gambarPath = path.join(userDir,
gambar.name);
        gambar_name = gambar.name;
        // console.log(gambarPath);
        await gambar.mv(gambarPath);
    }


    const response = await
UserModel.findOneAndUpdate({ _id }, { $set:
```

```
{ nama, alamat, gambar: gambar_name } }, {
new: true });
    console.log(response);
    res.status(200).json(response);
})


router.get('/user/kirim-paket', async (req:
Request, res: Response) => {
    res.sendFile(__dirname +
'/user_after_login_ui/paket.html');
})

router.post('/user/kirim-paket', async (req:
Request, res: Response) => {
    const { _id, no_telpon_pengirim,
nama_pengirim, alamat_pengirim,
no_telpon_penerima, nama_penerima,
alamat_penerima } = req.body;
    const { gambar_paket } = req.files as {
gambar_paket: UploadedFile };

    const userNoTelpon = (await
UserModel.findOne({ _id
}).select('no_telpon'))?.no_telpon;
    // console.log(userNoTelpon);

    const newKiriman = new KirimanModel({
id_pengirim: _id, no_telpon_pengirim,
nama_pengirim, alamat_pengirim,
no_telpon_penerima, nama_penerima,
alamat_penerima, gambar_paket:
gambar_paket.name });
    await newKiriman.save();
    // console.log(newKiriman);

    const gambarPath = path.join(__dirname,
'../images/kiriman/' + newKiriman._id + '/' +
gambar_paket.name);
    await gambar_paket.mv(gambarPath);

    const message = `Pengiriman anda
dengan nomor resi ${newKiriman._id}
sedang diproses oleh admin.`;

    try {
        await
axios.post('http://localhost:3012/send-
message', { number: userNoTelpon,
message: message });
    } catch (error) {
        console.log(error);
    }


socket_client.emit('pengiriman_baru_server');

    res.status(200).json("ini kirim paket")
})

router.get("/user/paket/:id/:status", async
(req: Request, res: Response) => {
    try {
        const { id, status } = req.params;
        console.log(id);
        let response
        if (status == "paket") {
            // check where status is not
"Dibatalkan Oleh Admin"
            response = await KirimanModel.find({
                id_pengirim: id,
                status: {
                    $nin: ['Dibatalkan Oleh Admin',
'Diterima Terverifikasi',"Dibatalkan Oleh
Pengirim"],
                    $type: 'string'
                }
            });
        } else {
            response = await KirimanModel.find({
id_pengirim: id });
        }
        console.log(response);
        res.status(200).json(response);
    } catch (error) {
        console.log(error);
        res.status(400).json([]);
    }

})

router.get("/user/paket-detail/:id", async (req:
Request, res: Response) => {
```

```
    const { id } = req.params;
    // console.log(id);
    try {
        const response = await
KirimanModel.findOne({ _id: id }).lean();
        if (!response) {
            res.status(404).json("Data tidak
ditemukan");
            return;
        }
        let responseWithKurir = {};
        if (response.id_kurir) {
            const kurirData = await
KurirModel.findOne({ _id: response.id_kurir
});


            responseWithKurir = {
                ...response,
                nama_kurir: kurirData?.nama,
                no_telpon_kurir:
kurirData?.no_telpon
            };
            // console.log(responseWithKurir);
        }
        // await 2 sec
        // await new Promise((resolve) =>
setTimeout(resolve, 2000));
        res.status(200).json(response.id_kurir ?
responseWithKurir : response);
    } catch (error) {
        console.log(error);
        res.status(400).json("error menampilkan
data");
    }
})


router.get("/user/paket-detail/:id/:gambar",
async (req: Request, res: Response) => {
    let { id, gambar } = req.params;
    // console.log(id , gambar);
    try {
        const response = await
KirimanModel.findOne({ _id: id,
gambar_paket: gambar });
        const gambarPath =
path.join(__dirname, '../images/kiriman/' + id
+ '/' + gambar);
        res.sendFile(gambarPath);
        // await 2 sec
        // await new Promise((resolve) =>
setTimeout(resolve, 2000));
        // res.status(200).json(response);
    } catch (error) {
        console.log(error);
        res.status(400).json("error menampilkan
data");
    }
    // res.status(200).json("success");

})



router.get('/user/profil', (req: Request, res:
Response) => {
    res.sendFile(__dirname +
'/user_after_login_ui/profil.html');
})


router.post('/user/check', async (req:
Request, res: Response) => {
    if (!req.body) {
        res.status(400).json('Bad Request');
        return;
    }
    const { _id, no_telpon, password,
createdAt } = req.body;

    // console.log(_id , no_telpon , password ,
createdAt);
    try {
        // check the user by _id , no_telpon ,
password , createdAt
        const user = await UserModel.findOne({
_id, no_telpon, password, createdAt });
        if (!user) {
            res.status(400).json('User not found');
            return;
        }
```

```
        res.status(200).json('Success');
        return;
    } catch (error) {
        console.log(error);
        res.status(500).json('Terjadi Kesalahan
Server');
        return
    }

})

router.get('/user/cek-kurir/:id', async (req:
Request, res: Response) => {
    const { id } = req.params;
    console.log(id);

    if (!id) {
        res.status(400).json('Bad Request');
        return;
    }
    // res.status(200).json('Success');
    try {
        const kurir = await KurirModel.findOne({
_id: id });
        // console.log(kurir);
        if (!kurir) {
            res.status(400).json('Kurir not found');
            return;
        }
        res.status(200).json(kurir);
        return;
    } catch (error) {
        console.log(error);
        res.status(500).json('Terjadi Kesalahan
Server');
        return
    }
})


router.get('/user/kurir/gambar/:no_telpon/:jeni
s', (req: Request, res: Response) => {
    const { no_telpon, jenis } = req.params;
    if (!no_telpon) {
        res.status(400).json({ error: 'No Telpon
is required.' });
        return
    }
    KurirModel.findOne({ no_telpon
}).then((data) => {
        // if data not found
        if (!data) {
            res.status(404).json({ error: 'Kurir not
found.' });
            return
        }
        // Define file paths
        const motorGambar =
path.join(__dirname, '../images/' + jenis + '/' +
data.gambar_motor);
        const kurirGambar =
path.join(__dirname, '../images/kurir/' +
data.gambar_kurir);

        if (jenis == 'motor') {
            res.sendFile(motorGambar);
        } else {
            res.sendFile(kurirGambar);
        }
    })
})

router.post('/user/terima-paket/:id_kiriman',
async (req: Request, res: Response) => {
    const { id_kiriman } = req.params;
    const { id_pengirim } = req.body;
    console.log(id_kiriman, id_pengirim);

    if (!id_kiriman || !id_pengirim) {
        res.status(400).json('Bad Request');
        return;
    }

    // cek the kiriman by id : id_kiriman ,
id_pengirim : id_pengirim
    const kiriman = await
KirimanModel.findOne({ _id: id_kiriman,
id_pengirim });
    if (!kiriman) {
        res.status(400).json('Kiriman not
found');
        return;
    }
```

```
    try {
        const updatedDoc = await
KirimanModel.findOneAndUpdate({ _id:
id_kiriman }, {
            $set: {
                status: 'Diterima Terverifikasi'
            },
            $push: {
                timeline: {
                    status: 'Diterima Terverifikasi',
                    waktu: new Date(),
                    // alasan: alasan || null,
                    id_kurir: kiriman.id_kurir
                }
            }
        }, { new: true });
        console.log(updatedDoc, "diterima
terverifikasi");

socket_client.emit('diterima_terverifikasi',
updatedDoc);
        res.status(200).json('Success');
    } catch (error) {
        console.log(error);
        res.status(500).json('Terjadi Kesalahan
Server');
    }



})

router.delete('/user/delete-paket/:id_kiriman',
async (req: Request, res: Response) => {
    const { id_kiriman } = req.params;
    const { id_pengirim } = req.body;

    if (!id_kiriman || !id_pengirim) {
        res.status(400).json('Bad Request');
        return;
    }

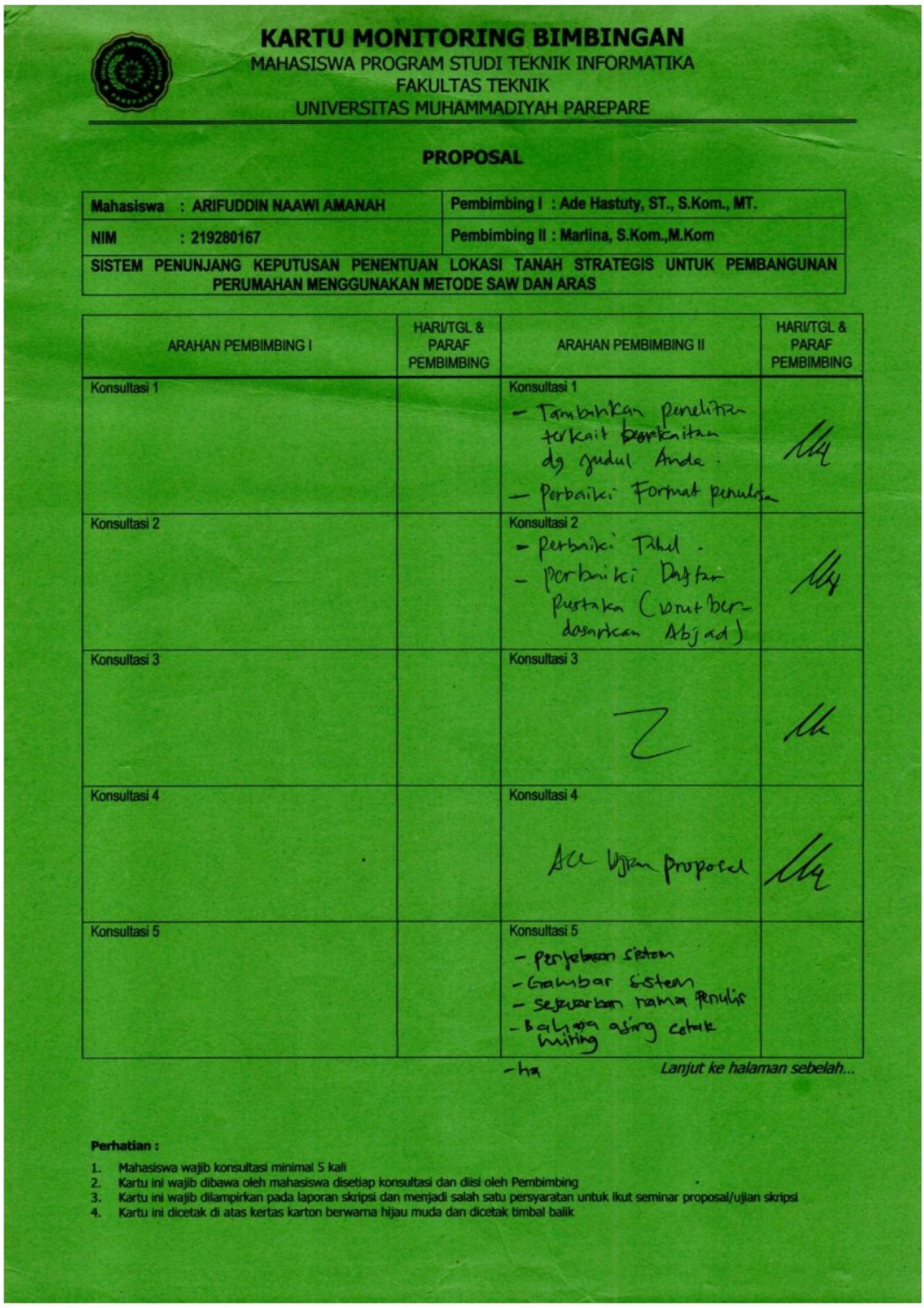
    // search by id : id_kiriman , id_pengirim :
id_pengirim
    const kirimanData =  await
KirimanModel.findOne({ _id: id_kiriman,
id_pengirim });

    if (!kirimanData) {
        res.status(400).json('Kiriman not
found');
        return;
    }

    try {
        // delete the kiriman
        const deletedDoc = await
KirimanModel.findOneAndUpdate({ _id:
id_kiriman }, {
            $set: {
                status: 'Dibatalkan Oleh Pengirim'
            },
            $push: {
                timeline: {
                    status: 'Dibatalkan Oleh
Pengirim',
                    waktu: new Date(),
                    // alasan: alasan || null,
                    id_kurir: kirimanData.id_kurir
                }
            }
        }, { new: true });
        socket_client.emit('delete_kiriman',
deletedDoc);
        res.status(200).json('Success');
    } catch (error) {
        console.log(error);
        res.status(500).json('Terjadi Kesalahan
Server');
    }
})


router.get('/user/history', async (req:
Request, res: Response) => {
    res.sendFile(__dirname +
'/user_after_login_ui/history.html');
})
// This should be the last route
router.use((req: Request, res: Response) =>
{
    res.status(404).sendFile(__dirname +
'/user_ui/404.html');
});
export default router;
```

Lampiran 2 – Kartu Monitoring Bimbingan Proposal



KARTU MONITORING BIMBINGAN
MAHASISWA PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS MUHAMMADIYAH PAREPARE

PROPOSAL

Mahasiswa : ARIFUDDIN NAAWI AMANAH	**Pembimbing I : Ade Hastuty, ST., S.Kom., MT.**
NIM : 219280167	**Pembimbing II : Marlina, S.Kom.,M.Kom**
SISTEM PENUNJANG KEPUTUSAN PENENTUAN LOKASI TANAH STRATEGIS UNTUK PEMBANGUNAN PERUMAHAN MENGGUNAKAN METODE SAW DAN ARAS	

ARAHAN PEMBIMBING I	HARI/TGL & PARAF PEMBIMBING	ARAHAN PEMBIMBING II	HARI/TGL & PARAF PEMBIMBING
Konsultasi 1		Konsultasi 1 - Tambahkan penelitian terkait bertkaitan dg judul Anda. - Perbaiki Format penulisan	
Konsultasi 2		Konsultasi 2 - perbaiki Tabel. - perbaiki Daftar Pustaka (urut berdasarkan Abjad)	
Konsultasi 3		Konsultasi 3	
Konsultasi 4		Konsultasi 4 Acc Ujian proposal	
Konsultasi 5		Konsultasi 5 - Perjelaskan sistem - Gambar sistem - Sejawatkan nama penulis - Bahasa asing cetak miring	

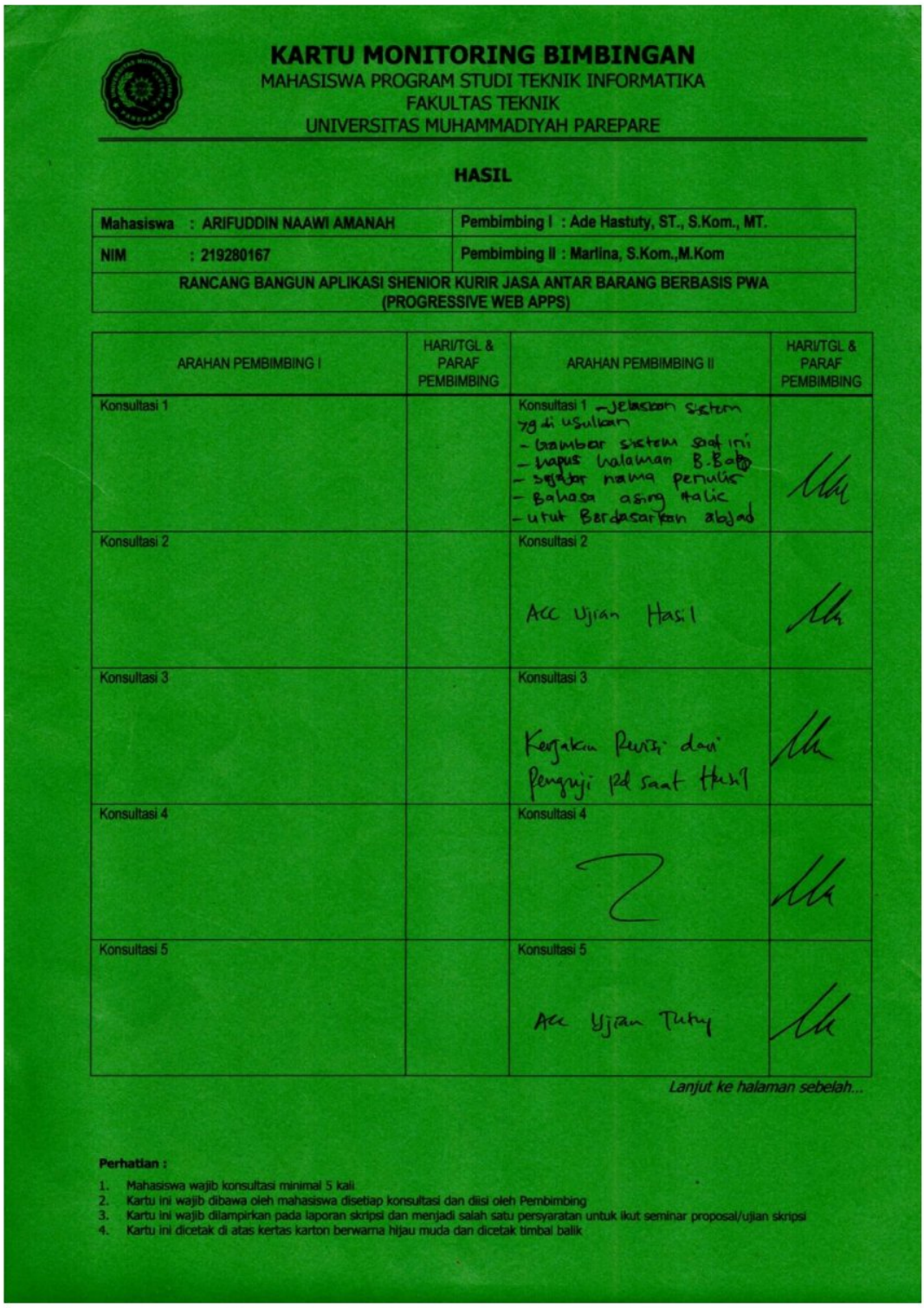
-ha

Lanjut ke halaman sebelah...

Perhatian :

1. Mahasiswa wajib konsultasi minimal 5 kali
2. Kartu ini wajib dibawa oleh mahasiswa disetiap konsultasi dan diisi oleh Pembimbing
3. Kartu ini wajib dilampirkan pada laporan skripsi dan menjadi salah satu persyaratan untuk ikut seminar proposal/ujian skripsi
4. Kartu ini dicetak di atas kertas karton berwarna hijau muda dan dicetak timbal balik

Lampiran 3 – Kartu Monitoring Bimbingan Hasil & Skripsi



KARTU MONITORING BIMBINGAN
MAHASISWA PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS MUHAMMADIYAH PAREPARE

HASIL

Mahasiswa : ARIFUDDIN NAAWI AMANAH	**Pembimbing I : Ade Hastuty, ST., S.Kom., MT.**
NIM : 219280167	**Pembimbing II : Marlina, S.Kom.,M.Kom**
RANCANG BANGUN APLIKASI SHENIOR KURIR JASA ANTAR BARANG BERBASIS PWA (PROGRESSIVE WEB APPS)	

ARAHAN PEMBIMBING I	HARI/TGL & PARAF PEMBIMBING	ARAHAN PEMBIMBING II	HARI/TGL & PARAF PEMBIMBING
Konsultasi 1		Konsultasi 1 - Jelaskan Sistem yg di usulkan - Gambar sistem saat ini - Hapus halaman B.Bab - sejarah nama penulis - Bahasa asing italic - urut Berdasarkan abjad	
Konsultasi 2		Konsultasi 2 ACC Ujian Hasil	
Konsultasi 3		Konsultasi 3 Kerjakan Revisi dari Penguji pd saat Hasil	
Konsultasi 4		Konsultasi 4	
Konsultasi 5		Konsultasi 5 Acc Ujian Tutup	

Lanjut ke halaman sebelah...

Perhatian :

1. Mahasiswa wajib konsultasi minimal 5 kali
2. Kartu ini wajib dibawa oleh mahasiswa disetiap konsultasi dan diisi oleh Pembimbing
3. Kartu ini wajib dilampirkan pada laporan skripsi dan menjadi salah satu persyaratan untuk ikut seminar proposal/ujian skripsi
4. Kartu ini dicetak di atas kertas karton berwarna hijau muda dan dicetak timbal balik

Lampiran 4 – Sertifikat Laboratorium Praktikum V



SERTIFIKAT

Nomor :517/PRK/LB-TI/II.3.AU/D/2023

diberikan kepada :

Arifuddin Naawi A

NIM.21928O167

telah menyelesaikan

PRAKTIKUM V

Laboratorium Teknik Informatika
Program Studi Teknik Informatika Fakultas Teknik
Universitas Muhammadiyah Parepare

Pada tanggal 16 Januari 2023 - 29 Maret 2023
Parepare 4/12/2023

KETUA PRODI
TEKNIK INFORMATIKA

MARLIAA.S.Kom., M.Kom

KEPALA
LABORATORIUM

Ir. UNTUNG SUWARDOYO, S.Kom., MT.,IPP.

Nilai Praktikum laboratorium Teknik Informatika
Fakultas Teknik Universitas Muhammadiyah Parepare Tahun 2023

MATERI	NILAI ANGKA
PENGOLAHAN CITRA DIGITAL	92
AUGMENTED DAN VIRTUAL REALITY	75
MIKROKONTROLER	70
PEMROGRAMAN MOBILE	90
RATA-RATA	82,87
NILAI HURUF	A