LAMPIRAN

1. index.js

```
// create express app
const express = require('express');
const cors = require('cors');
const fileUpload = require('express-
fileupload');
const dotenv = require('dotenv');
const path = require('path');
const conn = require('./conn');
const connection = conn.connection;
const app = express();

dotenv.config();

app.use(express.static(path.join(__dirnam
e, 'assets')));

app.use(cors());
app.use(express.json());
app.use(express.urlencoded({ extended:
true }));
app.use(fileUpload());

app.get('/', (req, res) => {
   console.log("buka halaman utama");
   res.sendFile(__dirname +
'/ui/index.html');

})

app.get('/jumlah_guru', async (req, res)
=> {
   const query = 'SELECT COUNT(*)
FROM tb_guru';
   try {
      const result = await new
Promise((resolve, reject) => {
         connection.query(query, (error,
results) => {
            if (error) {
               reject(error);
            } else {
               resolve(results);
            }
         });
      });

      return res.status(200).json({ success:
true, data: result[0]['COUNT(*)'], status:
true });
   } catch (error) {
      console.log(error);
      return res.status(500).json({
message: 'Internal server error', status:
false });
   }


})

app.get('/jumlah_siswa', async (req, res)
=> {
   const query = 'SELECT COUNT(*)
FROM tb_siswa';
   try {
      const result = await new
Promise((resolve, reject) => {
         connection.query(query, (error,
results) => {
            if (error) {
               reject(error);
            } else {
               resolve(results);
            }
         });
      });

      return res.status(200).json({ success:
true, data: result[0]['COUNT(*)'], status:
true });

   } catch (error) {
      console.log(error);
      return res.status(500).json({
message: 'Internal server error', status:
false });
   }
})

app.get('/jumlah_staf', async (req, res)
=> {
   const query = 'SELECT COUNT(*)
FROM tb_staf';
```

```
    try {
        const result = await new
Promise((resolve, reject) => {
            connection.query(query, (error,
results) => {
                if (error) {
                    reject(error);
                } else {
                    resolve(results);
                }
            });
        });

        return res.status(200).json({ success:
true, data: result[0]['COUNT(*)'], status:
true });

    } catch (error) {
        console.log(error);
        return res.status(500).json({
message: 'Internal server error', status:
false });
    }
})

app.use('/before-login',
require('./routes/before_login_route'));
app.use('/admin',
require('./routes/admin_route'));
app.use('/guru',
require('./routes/guru_route'));
app.use('/ortu',
require('./routes/ortu_route'));

// error handler
app.use((err, req, res, next) => {
    console.error(err.stack);
    res.status(500).send('Something
broke!');
})



const port = process.env.PORT || 3002;

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

2. admin_route.js

```
const express = require('express');
const router = express.Router();
const dotenv = require('dotenv');
const conn = require('../conn');
const connection = conn.connection;
const md5 = require('md5');
const path = require('path');
const fs = require('fs');
// connection.connect((err) => {
//     if (err) {
//         console.error('Error connecting to
MySQL database: ' + err.stack);
//         return;
//     }
//     console.log('Connected to MySQL
database as id ' + connection.threadId);
// })

dotenv.config();

router.get('/', async (req, res) => {
    res.sendFile(__dirname +
'/ui/admin/index.html');
})


// begin kelas
router.get('/kelas', async (req, res) => {
    res.sendFile(__dirname +
'/ui/admin/kelas.html');
})

router.get('/kelas/get', async (req, res)
=> {
    const query = 'SELECT * FROM
tb_kelas';
    connection.query(query, async (error,
results) => {
        if (error) {
```

```
            console.log('error get kelas',
error);
            return res.status(500).json({
message: 'Internal server error', status:
false });
        }
        let data = [];
        let i = 0;
        for (let i = 0; i < results.length; i++)
{
            let nip_guru = results[i].nip_guru
== null ? '' : results[i].nip_guru

            const query_wali_kelas = 'SELECT
* FROM tb_guru WHERE nip = ?';
            const result_wali_kelas = await
new Promise((resolve, reject) => {

connection.query(query_wali_kelas,
[nip_guru], (error, results) => {
                    if (error) {
                        reject(error);
                    } else {
                        resolve(results);
                    }
                })
            })

            data[i] = {
                id_kelas: results[i].id_kelas,
                nama_kelas:
results[i].nama_kelas,
                nip_guru: nip_guru,
                nama_guru: nip_guru == '' ? '-'
: result_wali_kelas[0].nama
            }
        }

        let out = Object.values(data);

        // console.log(out)

        return res.status(200).json({ success:
true, data: out, status: true });
    })
})

router.post('/kelas', async (req, res) => {
    const { kelas ,wali} = req.body;
    const query = 'INSERT INTO tb_kelas
(nama_kelas, nip_guru) VALUES (?, ?)';
    connection.query(query, [kelas, wali],
(error, results) => {
        if (error) {
            console.log('error insert kelas',
error);
            return res.status(500).json({
message: 'Internal server error', status:
false });
        }
        return res.status(200).json({ success:
true, data: results, status: true });
    })
})

router.put('/kelas', async (req, res) => {
    const { id, kelas ,wali} = req.body;
    const query = 'UPDATE tb_kelas SET
nama_kelas = ? , nip_guru = ? WHERE
id_kelas = ?';
    connection.query(query, [kelas, wali,
id], (error, results) => {
        if (error) {
            console.log('error update kelas',
error);
            return res.status(500).json({
message: 'Internal server error', status:
false });
        }
        return res.status(200).json({ success:
true, data: results, status: true });
    })
})

router.delete('/kelas/:id', async (req, res)
=> {
    const { id } = req.params;
    const query = 'DELETE FROM tb_kelas
WHERE id_kelas = ?';
    connection.query(query, [id], (error,
results) => {
        if (error) {
            console.log('error delete kelas',
error);
```

```
            return res.status(500).json({
message: 'Internal server error', status:
false });
        }
        return res.status(200).json({ success:
true, data: results, status: true });
    })
})
// end kelas

// begin siswa
router.get('/siswa', async (req, res) => {
    res.sendFile(__dirname +
'/ui/admin/siswa.html');
})

router.get('/siswa/get', async (req, res)
=> {
    const query = 'SELECT * FROM
tb_siswa';

    try {
        const result = await new
Promise((resolve, reject) => {
            connection.query(query, (error,
results) => {
                if (error) {
                    reject(error);
                } else {
                    resolve(results);
                }
            });
        });

        let data = [];
        for (let i = 0; i < result.length; i++) {
            const result_data = result[i];

            // Perform the asynchronous
operation inside the loop
            const result_check_kelas = await
new Promise((resolve, reject) => {
                connection.query('SELECT *
FROM tb_kelas WHERE id_kelas = ?',
[result_data.id_kelas], (error, results) => {
                    if (error) {
                        reject(error);
                    } else {
                        resolve(results);
                    }
                });
            });

            data.push({
                nis: result_data.nis,
                nama: result_data.nama,
                angkatan:
result_data.angkatan,
                kelas: result_data.kelas,
                nama_kelas:
result_check_kelas.length > 0 ?
result_check_kelas[0].nama_kelas : ''
            });
        }

        let out = Object.values(data);

        console.log(out);

        return res.status(200).json({ success:
"ini di get all", data: out, status: true });
    } catch (error) {
        res.status(500).json({ message:
'Internal server error', status: false });
    }

    // connection.query(query, (error,
results) => {
    //     if (error) {
    //         console.log('error get siswa',
error);
    //         return res.status(500).json({
message: 'Internal server error', status:
false });
    //     }
    //     let data = [];
    //     let i = 0;
    //     results.forEach(result => {


    //         data[i] = {};
    //         data[i].nis = result.nis;
    //         data[i].nama = result.nama;
```

```
    //         data[i].angkatan =
result.angkatan;
    //         data[i].kelas = result.kelas;
    //         data[i].nama_kelas =
result.nama_kelas;
    //         i++;
    //     })

    //     let out = Object.values(data);

    //     // console.log(out)

    //     return res.status(200).json({
success: true, data: out, status: true });
    // })
})

router.get('/siswa/get/:nis', async (req,
res) => {
    // await 2 seconds
    const { nis } = req.params;
    console.log(nis)
    const query = 'SELECT
a.nis,a.nama,a.kelas,a.angkatan,a.foto,b.i
d_kelas,c.nik,c.tempat_lahir,c.tanggal_lahi
r,c.jenis_kelamin,c.agama,c.orang_tua_1,c
.orang_tua_2,c.no_hp_orang_tua,c.alamat
FROM tb_siswa a join tb_kelas b join
tb_detail_siswa c on
a.nama_kelas=b.id_kelas and a.nis=c.nis
where a.nis = ?';
    connection.query(query, [nis], (error,
results) => {
        if (error) {
            console.log('error get siswa',
error);
            return res.status(500).json({
message: 'Internal server error', status:
false });
        }
        console.log(results)
        return res.status(200).json({ success:
"ini di get idnya", data: results[0], status:
true });
    })
})

router.get('/siswa/gambar/:gambar',
async (req, res) => {
    const { gambar } = req.params;
    const dir = path.join(__dirname,
'../image/' + gambar);
    // console.log(dir)
    res.sendFile(dir);
})


router.post('/siswa', async (req, res) => {
    var { nis, nama, kelas, nama_kelas,
angkatan, nik, tempat_lahir,
tanggal_lahir, jenis_kelamin, agama,
orang_tua1, orang_tua2,
no_hp_orang_tua, alamat } = req.body;
    const { gambar } = req.files;
    const query_check_nik = 'SELECT *
FROM tb_detail_siswa WHERE nik = ?';
    nama_kelas = nama_kelas != "null" ?
nama_kelas : null;

    // console.log(nama_kelas,nik)

    connection.query(query_check_nik,
[nik], (error_check_nik, results_check_nik)
=> {
        if (error_check_nik) {
            console.log('error check nik',
error_check_nik);
            return res.status(500).json({
success: true, message: 'Internal server
error', status: true });
        }
        if (results_check_nik.length > 0) {
            return res.status(500).json({
success: true, message: 'NIK sudah
terdaftar', status: true });
        }
        // upload the image to 2 folders
outside of this folder
        // move the file to the 'image'
folder and change it to "nis.jpg"
        const destinationPath =
path.join(__dirname, '../image/',
`${nis}.jpg`);
```

```
        gambar.mv(destinationPath, (err)
=> {
            if (err) {
                console.log(err);
                return res.status(500).json({
success: true, data: {}, status: true });
            }

            console.log('File uploaded
successfully');
        })

        const query_insert_siswa = "INSERT
INTO tb_siswa(nis, nama, kelas,
nama_kelas, angkatan, foto) VALUES
(?,?,?,?,?,?)";


connection.query(query_insert_siswa,
[nis, nama, kelas, nama_kelas, angkatan,
'image/' + `${nis}.jpg`],
(error_insert_siswa, results_insert_siswa)
=> {
            if (error_insert_siswa) {
                console.log('error insert siswa',
error_insert_siswa);
                if (error_insert_siswa.code ===
'ER_DUP_ENTRY') {
                    return res.status(500).json({
success: true, message: 'NIS sudah
terdaftar', status: true });
                }
                return res.status(500).json({
success: true, message:
error_insert_siswa, status: true });
            }
            const query_insert_detail_siswa =
"INSERT INTO tb_detail_siswa (nis,nik,
tempat_lahir, tanggal_lahir,jenis_kelamin,
agama, orang_tua_1, orang_tua_2,
no_hp_orang_tua, alamat) VALUES
(?,?,?,?,?,?,?,?,?,?)";


connection.query(query_insert_detail_sis
wa, [nis, nik, tempat_lahir, tanggal_lahir,
jenis_kelamin, agama, orang_tua1,
orang_tua2, no_hp_orang_tua, alamat],
(error_insert_detail, results_insert_detail)
=> {
                if (error_insert_detail) {
                    console.log('error insert
detail siswa', error_insert_detail);
                    if (error_insert_detail.code
=== 'ER_DUP_ENTRY') {
                        return
res.status(500).json({ success: true,
message: 'NIK sudah terdaftar', status:
true });
                    }
                    return res.status(500).json({
success: true, message:
error_insert_detail, status: true });
                }
                const query_insert_user =
"INSERT INTO tb_user
(username,password,level,nis) values
(?,?,?,?)";
                const md5_password =
md5(nis); //ecrypted converted nis to
string

connection.query(query_insert_user, [nis,
md5_password, 'ortu', nis], (error,
results_insert_user) => {
                    if (error) {
                        console.log('error insert
user', error);
                        return
res.status(500).json({ success: true,
message: error, status: true });
                    }
                    return res.status(200).json({
success: true, message: 'Data siswa
tersimpan', status: true });
                })

                // return res.status(500).json({
success: true, data: results_insert_detail,
status: true });
            })
            // return res.status(500).json({
success: true, data: results_insert_siswa,
status: true });
```

```
        })

    })


    // return res.status(500).json({ success:
true, data: {}, status: true });
})


router.put('/siswa', (req, res) => {

    const { nis, nama, kelas, nama_kelas,
angkatan, nik, tempat_lahir,
tanggal_lahir, jenis_kelamin, agama,
alamat, orang_tua_1, orang_tua_2,
no_hp_orang_tua, nis_lama } = req.body;
    let gambar = null;
    if (req.files) {
        gambar = req.files.gambar
        // console.log(gambar)

        // upload the image to 2 folders
outside of this folder
        // move the file to the 'image'
folder and change it to "nis.jpg"
        const destinationPath =
path.join(__dirname, '../image/',
`${nis}.jpg`);
        gambar.mv(destinationPath, (err)
=> {
            if (err) {
                console.log(err);
                return res.status(500).json({
success: true, data: {}, status: true });
            }

            console.log('File uploaded
successfully');
        })
    }

    const query_update_siswa = "UPDATE
tb_siswa SET nis = ?, nama = ?, kelas = ?,
nama_kelas = ?, angkatan = ?, foto = ?
WHERE nis = ?";

    connection.query(query_update_siswa,
[nis, nama, kelas, nama_kelas, angkatan,
'image/' + `${nis}.jpg`, nis_lama],
(error_update_siswa,
results_update_siswa) => {
        if (error_update_siswa) {
            console.log('error update siswa',
error_update_siswa);
            return res.status(500).json({
success: true, message:
error_update_siswa.message, status: true
});
        }
        if (nis != nis_lama && !req.files) {
            // move the file from
'image/nis_lama.jpg' to 'image/nis.jpg'
            const sourcePath =
path.join(__dirname, '../image/',
`${nis_lama}.jpg`);
            const destinationPath =
path.join(__dirname, '../image/',
`${nis}.jpg`);

            fs.rename(sourcePath,
destinationPath, (err) => {
                if (err) {
                    console.log(err);
                    return res.status(500).json({
success: true, data: {}, status: true });
                }

                console.log('File moved
successfully');
            })

        }

        const query_update_detail =
"UPDATE tb_detail_siswa SET nik = ?,
tempat_lahir = ?, tanggal_lahir = ?,
jenis_kelamin = ?, agama =
?,orang_tua_1 = ?, orang_tua_2 = ?,
no_hp_orang_tua = ?, alamat = ? WHERE
nis = ?";


connection.query(query_update_detail,
```

```
[nik, tempat_lahir, tanggal_lahir,
jenis_kelamin, agama, orang_tua_1,
orang_tua_2, no_hp_orang_tua, alamat,
nis], async (error_update_detail,
results_update_detail) => {
        if (error_update_detail) {
            console.log('error update
detail siswa', error_update_detail);
            return res.status(500).json({
success: true, message:
error_update_detail.message, status: true
});
        }
        if (nis != nis_lama) {
            const query_login = "UPDATE
tb_user SET username = ?, password = ?
WHERE nis = ?";
            await new Promise((resolve,
reject) => {

connection.query(query_login, [nis,
md5(nis), nis], (error, results) => {
                    if (error) {
                        reject(error);
                    } else {
                        resolve(results);
                    }
                });
            })
        }
        return res.status(200).json({
success: true, message: 'Data siswa
tersimpan', status: true });
    })
    // console.log(results_update_siswa)

    // return res.status(500).json({
success: true, message: 'Data siswa
tersimpan', status: true });
  })



  // res.status(500).json({ success: true,
message: 'Error tidak diketahui', status:
true });
})


router.delete('/siswa/:nis', (req, res) => {

  const { nis } = req.params;

  const query_delete_siswa = "DELETE
FROM tb_siswa WHERE nis = ?";
  connection.query(query_delete_siswa,
[nis], (error_delete_siswa,
results_delete_siswa) => {
    if (error_delete_siswa) {
        console.log('error delete siswa',
error_delete_siswa);
        return res.status(500).json({
success: true, message:
error_delete_siswa.message, status: true
});
    }
    // delete teh image
    const sourcePath =
path.join(__dirname, '../image/',
`${nis}.jpg`);

    fs.unlink(sourcePath, (err) => {
        if (err) {
            console.log(err);
            return res.status(500).json({
success: true, data: {}, status: true });
        }

        console.log('File deleted
successfully');
    })

    return res.status(200).json({ success:
true, message: 'Data siswa terhapus',
status: true });
  })



})

// end siswa
```

```
// begin guru
router.get('/guru', async (req, res) => {
    res.sendFile(__dirname +
'/ui/admin/guru.html');
})

router.get('/guru/get', async (req, res)
=> {
    const query = "SELECT * FROM
tb_guru";
    connection.query(query, (error,
results) => {
        if (error) {
            console.log('error get guru',
error);
            return res.status(500).json({
success: true, data: {}, status: true });
        }
        return res.status(200).json({ success:
true, data: results, status: true });
    })
})

router.get('/guru/get/:nip', async (req,
res) => {
    const { nip } = req.params;
    const query = "SELECT * FROM
tb_guru a join tb_detail_guru b on a.nip
= b.nip WHERE a.nip = ?";
    connection.query(query, [nip], (error,
results) => {
        if (error) {
            console.log('error get guru',
error);
            return res.status(500).json({
success: true, data: {}, status: true });
        }
        return res.status(200).json({ success:
true, data: results[0], status: true });
    })
})

router.get('/guru/gambar/:gambar',
async (req, res) => {
    const { gambar } = req.params;
    const dir = path.join(__dirname,
'../guru_image/' + gambar);
    // console.log(dir)
    res.sendFile(dir);
})

router.put('/guru', async (req, res) => {
    const { nip, nama, matpel, nik,
tempat_lahir, tanggal_lahir,
jenis_kelamin, agama, no_hp, alamat,
nip_lama } = req.body;
    let gambar = null;
    if (req.files) {
        gambar = req.files.gambar;

        const destinationPath =
path.join(__dirname, '../guru_image/',
`${nip}.jpg`);
        gambar.mv(destinationPath, (err)
=> {
            if (err) {
                console.log(err);
                // return res.status(500).json({
success: true, data: {}, status: true });
            }
            console.log('File moved
successfully');
        })
    }

    let today_date_time = new Date();
    // if the month is less than 10 then
add 0 before it
    let date =
today_date_time.getFullYear() + '-' +
(today_date_time.getMonth() < 10 ? '0' +
(today_date_time.getMonth() + 1) :
(today_date_time.getMonth() + 1)) + '-'
+ today_date_time.getDate();

    const query_update_guru = "UPDATE
tb_guru SET nip = ?, nama = ?, matpel =
? , updated_at = ? WHERE nip = ?";
    connection.query(query_update_guru,
[nip, nama, matpel, date, nip_lama],
(error_update_guru,
results_update_guru) => {
        if (error_update_guru) {
```

```
            console.log('error update guru',
error_update_guru);
            return res.status(500).json({
success: true, message:
error_update_guru.message, status: true
});
        }

        if (nip != nip_lama && !req.files) {
            const sourcePath =
path.join(__dirname, '../guru_image/',
`${nip_lama}.jpg`);
            const destinationPath =
path.join(__dirname, '../guru_image/',
`${nip}.jpg`);

            fs.copyFile(sourcePath,
destinationPath, (err) => {
                if (err) {
                    console.log(err);
                    return res.status(500).json({
success: true, data: {}, status: true });
                }
                console.log('File copied
successfully');
            })
        }

        const query_update_detail_guru =
"UPDATE tb_detail_guru SET nik = ?,
tempat_lahir = ?, tanggal_lahir = ?,
jenis_kelamin = ?, agama = ?, no_hp = ?,
alamat = ? ,updated_at = ? WHERE nip =
?";


connection.query(query_update_detail_g
uru, [nik, tempat_lahir, tanggal_lahir,
jenis_kelamin, agama, no_hp, alamat,
date, nip], async
(error_update_detail_guru,
results_update_detail_guru) => {
            if (error_update_detail_guru) {
                console.log('error update
detail guru', error_update_detail_guru);
                return res.status(500).json({
success: true, message:
error_update_detail_guru.message,
status: true });
            }

            if (nip != nip_lama) {
                const query_update_login =
"UPDATE tb_user SET username = ?
WHERE nip = ?";
                await new Promise((resolve,
reject) => {

connection.query(query_update_login,
[nip, nip], (error, results) => {
                        if (error) {
                            console.log(error);
                            reject(error);
                        } else {
                            resolve(results);
                        }
                    })
                })
            }


            return res.status(200).json({
success: true, data:
results_update_detail_guru, status: true
});
            // return res.status(200).json({
success: true, data:
results_update_detail_guru, status: true
});
        })
    })
})

router.post('/guru', async (req, res) => {
    const { nip, nama, matpel, nik,
tempat_lahir, tanggal_lahir,
jenis_kelamin, agama, no_hp, alamat } =
req.body;
    const { gambar } = req.files;
    // console.log(gambar);
    // console.log(req.body)
    const destinationPath =
path.join(__dirname, '../guru_image/',
`${nip}.jpg`);
```

```
    gambar.mv(destinationPath, (err) => {
        if (err) {
            console.log(err);
            // return res.status(500).json({
success: true, data: {}, status: true });
        }

        console.log('File moved
successfully');
    })

    const query_insert_guru = "INSERT
INTO tb_guru (nip, nama, matpel)
VALUES (?, ?, ?)";

    connection.query(query_insert_guru,
[nip, nama, matpel], (error_insert_guru,
results_insert_guru) => {
        if (error_insert_guru) {
            console.log('error insert guru',
error_insert_guru);
            return res.status(500).json({
success: true, message:
error_insert_guru.message, status: true });
        }
        const query_insert_detail = "INSERT
INTO tb_detail_guru (nip, nik,
tempat_lahir, tanggal_lahir,jenis_kelamin,
agama, no_hp, alamat) VALUES (?, ?, ?, ?,
?, ?, ?, ?)";


connection.query(query_insert_detail,
[nip, nik, tempat_lahir, tanggal_lahir,
jenis_kelamin, agama, no_hp, alamat],
(error_insert_detail, results_insert_detail)
=> {
            if (error_insert_detail) {
                console.log('error insert detail
guru', error_insert_detail);
                return res.status(500).json({
success: true, message:
error_insert_detail.message, status: true
});
            }
            const query_insert_user =
"INSERT INTO tb_user (username,
password, level,nip) VALUES (?, ?, ?, ?)";


connection.query(query_insert_user, [nip,
md5(nip), 'guru', nip], (error_insert_user,
results_insert_user) => {
                if (error_insert_user) {
                    console.log('error insert
user', error_insert_user);
                    return res.status(500).json({
success: true, message:
error_insert_user.message, status: true });
                }

                return res.status(200).json({
success: true, message: 'Data guru
tersimpan', status: true });
            })
        })
    })
})

router.delete('/guru/:nip', async (req, res)
=> {

    const { nip } = req.params;

    const query_delete_guru = "DELETE
FROM tb_guru WHERE nip = ?";

    connection.query(query_delete_guru,
[nip], (error_delete_guru,
results_delete_guru) => {

        if (error_delete_guru) {
            console.log('error delete guru',
error_delete_guru);
            return res.status(500).json({
success: true, message:
error_delete_guru.message, status: true
});
        }
```

```
        const sourcePath =
path.join(__dirname, '../guru_image/',
`${nip}.jpg`);

        fs.unlink(sourcePath, (err) => {
            if (err) {
                console.log(err);
                return res.status(500).json({
success: true, data: {}, status: true });
            }
            console.log('File deleted
successfully');
        })

        return res.status(200).json({ success:
true, message: 'Data Guru Terhapus',
status: true });
    })
})

router.get('/guru/absen/:date', async
(req, res) => {
    const query = "SELECT * FROM
tb_guru";
    const { date } = req.params;
    console.log(date)
    connection.query(query, async (error,
results) => {
        if (error) {
            console.log('error get guru',
error);
            return res.status(500).json({
success: true, data: {}, status: true });
        }
        let data = [];
        let status_absen = false
        for (let index = 0; index <
results.length; index++) {
            const element = results[index];

            const check_absensi = "SELECT *
FROM tb_absensi_guru WHERE nip = ?
AND tanggal = ?";

            const result_absen = await new
Promise((resolve, reject) => {
                connection.query(check_absensi,
[element.nip, date], (error, results) => {
                    if (error) {
                        reject(error);
                    } else {
                        resolve(results);
                    }
                })
            });
            //
console.log(result_absen[0].absensi)
            var absensi = result_absen.length
> 0 ? result_absen[0].absensi : 1
            status_absen =
result_absen.length > 0 ? true : false

            data.push({
                nip: element.nip,
                nama: element.nama,
                absen: absensi
            })

        }

        let out = Object.values(data)
        return res.status(200).json({ success:
true, data: out, status: status_absen });
    })
})

router.post('/guru/absen', async (req,
res) => {
    const { data, tanggal, status } =
req.body
    // console.log(tanggal)
    var today = new Date(tanggal)
    var dd =
String(today.getDate()).padStart(2, '0');
    var mm = String(today.getMonth() +
1).padStart(2, '0'); //January is 0!
    var yyyy = today.getFullYear();
    today = yyyy + '-' + mm + '-' + dd;
    // console.log(today)
    for (let index = 0; index < data.length;
index++) {
        const element = data[index];
```

```
        const { nip, absen } = element
        const query_update_absen = status
== false ? "Insert into tb_absensi_guru
(absensi,tanggal,nip) values(?,?,?)" :
"Update tb_absensi_guru set absensi = ?
where tanggal = ? and nip = ?";
        const result = await
connection.query(query_update_absen,
[absen, today, nip]);
    }
    return res.status(200).json({ success:
true, data: data, status: true });
})

// end guru

// begin matapelajaran
router.get('/mat-pel', async (req, res) =>
{
    res.sendFile(__dirname +
'/ui/admin/mat_pel.html');
})

router.get('/mat-pel/get', async (req, res)
=> {

    const query_get_matpel = "SELECT *
FROM tb_matpel";

    connection.query(query_get_matpel,
(error_get_matpel, results_get_matpel)
=> {
        if (error_get_matpel) {
            console.log('error get matpel',
error_get_matpel);
            return res.status(500).json({
success: true, message:
error_get_matpel.message, status: true });
        }

        return res.status(200).json({ success:
true, data: results_get_matpel, status:
true });
    })
})

router.post('/mat-pel', async (req, res)
=> {
    const { matpel } = req.body;

    const query_insert_matpel = "INSERT
INTO tb_matpel (matpel) VALUES (?)";


connection.query(query_insert_matpel,
[matpel], (error_insert_matpel,
results_insert_matpel) => {
        if (error_insert_matpel) {
            console.log('error insert matpel',
error_insert_matpel);
            return res.status(500).json({
success: true, message:
error_insert_matpel.message, status: true
});
        }
        return res.status(200).json({ success:
true, message: 'Mata Pelajaran
Tersimpan', status: true });
    })
})

router.put('/mat-pel/', async (req, res)
=> {

    const { id_matpel, matpel } =
req.body;

    const query_update_matpel =
"UPDATE tb_matpel SET matpel = ?
WHERE id_matpel = ?";


connection.query(query_update_matpel,
[matpel, id_matpel],
(error_update_matpel,
results_update_matpel) => {
        if (error_update_matpel) {
            console.log('error update
matpel', error_update_matpel);
            return res.status(500).json({
success: true, message:
error_update_matpel.message, status:
true });
```

```
        }

        return res.status(200).json({ success:
true, message: 'Mata Pelajaran
Tersimpan', status: true });
    })
})

router.delete('/mat-pel/:id_matpel',
async (req, res) => {

    const { id_matpel } = req.params;

    const query_delete_matpel = "DELETE
FROM tb_matpel WHERE id_matpel = ?";


connection.query(query_delete_matpel,
[id_matpel], (error_delete_matpel,
results_delete_matpel) => {
        if (error_delete_matpel) {
            console.log('error delete matpel',
error_delete_matpel);
            return res.status(500).json({
success: true, message:
error_delete_matpel.message, status:
true });
        }
        return res.status(200).json({ success:
true, message: 'Mata Pelajaran
Terhapus', status: true });
    })
})

// end matapelajaran

// begin staf
router.get('/staf', async (req, res) => {
    res.sendFile(__dirname +
'/ui/admin/staf.html');
})

router.get('/staf/get', async (req, res) =>
{
    const query = "SELECT * FROM
tb_staf";

    connection.query(query, (error,
results) => {
        if (error) {
            console.log('error get staf', error);
            return res.status(500).json({
success: true, data: {}, status: true });
        }
        return res.status(200).json({ success:
true, data: results, status: true });
    })
})

router.get('/staf/get/:nip', async (req, res)
=> {

    const { nip } = req.params;
    const query = "SELECT * FROM tb_staf
WHERE nip = ?";

    connection.query(query, [nip], (error,
results) => {
        if (error) {
            console.log('error get staf', error);
            return res.status(500).json({
success: true, data: {}, status: true });
        }
        return res.status(200).json({ success:
true, data: results[0], status: true });
    })
})

router.get('/staf/gambar/:gambar', async
(req, res) => {

    const { gambar } = req.params;
    const dir = path.join(__dirname,
'../staf_image/' + gambar);
    res.sendFile(dir)
})

router.put('/staf', async (req, res) => {

    const { nip, nama, pekerjaan, nik,
tempat_lahir, tanggal_lahir,
jenis_kelamin, agama, no_hp, alamat,
nip_lama } = req.body;
    let gambar = null
```

```
    if (req.files) {
        gambar = req.files.gambar

        const destinationPath =
path.join(__dirname, '../staf_image/',
`${nip}.jpg`);
        gambar.mv(destinationPath, (err)
=> {
            if (err) {
                console.log(err);
                // return res.status(500).json({
success: true, data: {}, status: true });
            }
            console.log('File moved
successfully');
        })
    }



    let today_date_time = new Date();

    let date =
today_date_time.getFullYear() + '-' +
(today_date_time.getMonth() < 10 ? '0' +
(today_date_time.getMonth() + 1) :
(today_date_time.getMonth() + 1)) + '-'
+ today_date_time.getDate();

    const query_update_staf = "UPDATE
tb_staf SET nip = ?, nama = ?, pekerjaan
= ?, nik = ?, tempat_lahir = ?,
tanggal_lahir = ?, jenis_kelamin = ?,
agama = ?, no_hp = ?, alamat = ?
,updated_at = ? WHERE nip = ?";

    connection.query(query_update_staf,
[nip, nama, pekerjaan, nik, tempat_lahir,
tanggal_lahir, jenis_kelamin, agama,
no_hp, alamat, date, nip_lama],
(error_update_staf, results_update_staf)
=> {
        if (error_update_staf) {
            console.log('error update staf',
error_update_staf);
            return res.status(500).json({
success: true, message:
error_update_staf.message, status: true
});
        }
        if (nip != nip_lama && !req.files) {
            const sourcePath =
path.join(__dirname, '../staf_image/',
`${nip_lama}.jpg`);
            const destinationPath =
path.join(__dirname, '../staf_image/',
`${nip}.jpg`);

            fs.copyFile(sourcePath,
destinationPath, (err) => {
                if (err) {
                    console.error(err);
                    return res.status(500).json({
success: true, message: err.message,
status: true });
                }
                console.log('File copied
successfully');
            })
        }

        if (nip != nip_lama && req.files) {
            const sourcePath =
path.join(__dirname, '../staf_image/',
`${nip_lama}.jpg`);
            fs.unlink(sourcePath, (err) => {
                if (err) {
                    console.log(err);
                    return res.status(500).json({
success: true, data: {}, status: true });
                }
                console.log('File deleted
successfully');
            })
        }

        return res.status(200).json({ success:
true, message: 'Staf Tersimpan', status:
true });
    })
})

router.delete('/staf/:nip', async (req, res)
=> {
```

```
    const { nip } = req.params;

    const query_delete_staf = "DELETE
FROM tb_staf WHERE nip = ?";

    connection.query(query_delete_staf,
[nip], (error_delete_staf,
results_delete_staf) => {
        if (error_delete_staf) {
            console.log('error delete staf',
error_delete_staf);
            return res.status(500).json({
success: true, message:
error_delete_staf.message, status: true });
        }

        const sourcePath =
path.join(__dirname, '../staf_image/',
`${nip}.jpg`);

        fs.unlink(sourcePath, (err) => {
            if (err) {
                console.log(err);
                return res.status(500).json({
success: true, data: {}, status: true });
            }
            console.log('File deleted
successfully');
        })

        return res.status(200).json({ success:
true, message: 'Staf Terhapus', status:
true });
    })

})

router.post('/staf', async (req, res) => {

    const { nip, nama, pekerjaan, nik,
tempat_lahir, tanggal_lahir,
jenis_kelamin, agama, no_hp, alamat } =
req.body;
    const gambar = req.files.gambar;
    const destinationPath =
path.join(__dirname, '../staf_image/',
`${nip}.jpg`);

    gambar.mv(destinationPath, (err) => {
        if (err) {
            console.log(err);
            // return res.status(500).json({
success: true, data: {}, status: true });
        }

        console.log('File moved
successfully');
    })

    const query_insert_staf = "INSERT
INTO tb_staf (nip, nama, pekerjaan, nik,
tempat_lahir, tanggal_lahir,
jenis_kelamin, agama, no_hp, alamat)
VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?, ?)";

    connection.query(query_insert_staf,
[nip, nama, pekerjaan, nik, tempat_lahir,
tanggal_lahir, jenis_kelamin, agama,
no_hp, alamat], (error_insert_staf,
results_insert_staf) => {
        if (error_insert_staf) {
            console.log('error insert staf',
error_insert_staf);
            if (error_insert_staf.code ==
'ER_DUP_ENTRY') {
                return res.status(500).json({
success: true, message: 'NIP sudah ada',
status: true });
            }
            return res.status(500).json({
success: true, message:
error_insert_staf.message, status: true });
        }
        return res.status(200).json({ success:
true, message: 'Data Tersimpan', status:
true });
    })

})
// end staf
```

```
// begin dana bos
router.get('/dana_bos', async (req, res)
=> {
    res.sendFile(__dirname +
'/ui/admin/dana_bos.html');
})

router.get('/dana_bos/get', async (req,
res) => {

    const query = 'SELECT * FROM
tb_dana_bos order by tanggal desc';
    try {
        const result = await new
Promise((resolve, reject) => {
            connection.query(query, (error,
results) => {
                if (error) {
                    console.log(error);
                    reject(error);
                } else {
                    resolve(results);
                }
            })
        })

        return res.status(200).json({ success:
true, data: result, status: true });
    } catch (error) {
        console.log(error);
        return res.status(500).json({ success:
true, message: error.message, status:
true });
    }
})

router.get('/dana_bos/get/:id_dana_bos',
async (req, res) => {

    const { id_dana_bos } = req.params;
    console.log(id_dana_bos);
    const query = 'SELECT * FROM
tb_dana_bos WHERE id_dana_bos = ?';

    try {
        const result = await new
Promise((resolve, reject) => {
            connection.query(query,
[id_dana_bos], (error, results) => {
                if (error) {
                    console.log(error);
                    reject(error);
                } else {
                    resolve(results);
                }
            })
        })

        console.log(result);

        return res.status(200).json({ success:
true, data: result[0], status: true });
    } catch (error) {
        console.log(error);
        return res.status(500).json({ success:
true, message: error.message, status:
true });
    }

})

router.put('/dana_bos', async (req, res)
=> {

    const { id_dana_bos, jumlah, status,
tanggal, ket } = req.body;
    console.log(id_dana_bos, jumlah,
status, tanggal, ket);
    try {
        const query = 'UPDATE
tb_dana_bos SET jumlah = ?, status = ?,
tanggal = ?, ket = ? , updated_at =
NOW() WHERE id_dana_bos = ?';
        const result = await new
Promise((resolve, reject) => {
            connection.query(query, [jumlah,
status, tanggal, ket, id_dana_bos], (error,
results) => {
                if (error) {
                    console.log(error);
                    reject(error);
                } else {
                    resolve(results);
                }
```

```
            })
        })
        console.log(result);

        return res.status(200).json({ success:
true, message: 'Data Tersimpan', status:
true });
    } catch (error) {
        console.log(error);
        return res.status(500).json({ success:
true, message: error.message, status:
true });
    }
})

router.delete('/dana_bos/:id_dana_bos',
async (req, res) => {

    const { id_dana_bos } = req.params;

    const query = 'DELETE FROM
tb_dana_bos WHERE id_dana_bos = ?';

    try {
        const result = await new
Promise((resolve, reject) => {
            connection.query(query,
[id_dana_bos], (error, results) => {
                if (error) {
                    console.log(error);
                    reject(error);
                } else {
                    resolve(results);
                }
            })
        })
        console.log(result);
        return res.status(200).json({ success:
true, message: 'Detail Dana Bos
Terhapus', status: true });
    } catch (error) {
        console.log(error);
        return res.status(500).json({ success:
true, message: error.message, status:
true });
    }
})

router.post('/dana_bos', async (req, res)
=> {

    const { jumlah, status, tanggal, ket } =
req.body;
    try {
        const query = 'INSERT INTO
tb_dana_bos (jumlah, status, tanggal,
ket) VALUES (?, ?, ?, ?)';
        const result = await new
Promise((resolve, reject) => {
            connection.query(query, [jumlah,
status, tanggal, ket], (error, results) => {
                if (error) {
                    console.log(error);
                    reject(error);
                } else {
                    resolve(results);
                }
            })
        })

        return res.status(200).json({ success:
true, message: 'Data Tersimpan', status:
true });
    } catch (error) {
        console.log(error);
        return res.status(500).json({ success:
true, message: error.message, status:
true });
    }
})

// end dana bos

// start surat
router.get('/surat', async (req, res) => {
    res.sendFile(__dirname +
'/ui/admin/surat.html');
})

router.get('/surat/get', async (req, res)
=> {

    const query = 'SELECT * FROM
tb_surat';
```

```
    try {
        const result = await new
Promise((resolve, reject) => {
            connection.query(query, (error,
results) => {
                if (error) {
                    console.log(error);
                    reject(error);
                } else {
                    resolve(results);
                }
            })
        })

        console.log(result);

        return res.status(200).json({ success:
true, data: result, status: true });
    } catch (error) {
        console.log(error);
        return res.status(500).json({ success:
true, message: error.message, status:
true });
    }
})

router.get('/surat/get/:id_surat', async
(req, res) => {

    const { id_surat } = req.params;
    const query = 'SELECT * FROM
tb_surat WHERE id_surat = ?';
    try {
        const result = await new
Promise((resolve, reject) => {
            connection.query(query,
[id_surat], (error, results) => {
                if (error) {
                    console.log(error);
                    reject(error);
                } else {
                    resolve(results);
                }
            })
        })

        return res.status(200).json({ success:
true, data: result[0], status: true });
    } catch (error) {
        console.log(error);
        return res.status(500).json({ success:
true, message: error.message, status:
true });
    }
})

router.get('/surat/pdf/:id_surat', async
(req, res) => {

    const { id_surat } = req.params;
    const dir = path.join(__dirname,
'../surat/' + id_surat + '.pdf');

    res.sendFile(dir)
})


router.post('/surat', async (req, res) => {
    const { status, tanggal, ket } =
req.body;
    const { pdf } = req.files;

    try {
        const query = 'INSERT INTO
tb_surat (status, tanggal, ket) VALUES (?,
?, ?)';
        const result = await new
Promise((resolve, reject) => {
            connection.query(query, [status,
tanggal, ket], (error, results) => {
                if (error) {
                    console.log(error);
                    reject(error);
                } else {
                    resolve(results);
                }
            })
        })
        console.log(result);
        const dir = path.join(__dirname,
'../surat/' + result.insertId + '.pdf');

        pdf.mv(dir, (err) => {
```

```
            if (err) {
                console.log(err);
            }
            console.log('File Uploaded');
        })

        return res.status(200).json({ success:
true, message: 'Data Tersimpan', status:
true });
    } catch (error) {
        console.log(error);
        return res.status(500).json({ success:
true, message: error.message, status:
true });
    }
})

router.delete('/surat/:id_surat', async
(req, res) => {

    const { id_surat } = req.params;

    const query = 'DELETE FROM tb_surat
WHERE id_surat = ?';

    try {
        const result = await new
Promise((resolve, reject) => {
            connection.query(query,
[id_surat], (error, results) => {
                if (error) {
                    console.log(error);
                    reject(error);
                } else {
                    resolve(results);
                }
            })
        })
        // delete pdf
        const dir = path.join(__dirname,
'../surat/' + id_surat + '.pdf');

        return res.status(200).json({ success:
true, message: 'Data Tersimpan', status:
true });

    } catch (error) {
        console.log(error);
        return res.status(500).json({ success:
true, message: error.message, status:
true });
    }
})


module.exports = router
```

3. guru_route.js

```
const express = require('express');
const router = express.Router();
const dotenv = require('dotenv');
const conn = require('../conn');
const connection = conn.connection;
// const md5 = require('md5');

dotenv.config();

router.get('/', async (req, res) => {
    // run the login file in parent folder /ui
    res.sendFile(__dirname +
'/ui/guru/index.html');
})

router.get('/mata_pelajaran_count', async
(req, res) => {
    let nip = req.query.nip
    // console.log(nip)
    const query = 'SELECT * from tb_guru
where nip = ?';
    try {
        const result = await new
Promise((resolve, reject) => {
            connection.query(query, [nip],
(error, results) => {
                if (error) {
                    reject(error);
                } else {
                    resolve(results);
                }
            })
        })
```

```
        if (result.length == 0) return
res.status(200).json({ success: true, data:
0, status: true });


        let list_mata_pelajaran = [];
        let list_kode_mapel =
result[0].matpel;
        console.log(list_kode_mapel)
        if (list_kode_mapel != null ||
list_kode_mapel != undefined ||
list_kode_mapel != "") {
            list_kode_mapel =
JSON.parse(list_kode_mapel);
        }
        const mata_pelajaran_counter =
list_kode_mapel.length || 0;
        for (let i = 0; i <
list_kode_mapel.length; i++) {

console.log(list_kode_mapel[i],"ini dia")
            let query_mata_pelajaran =
'SELECT * from tb_matpel where
id_matpel = ?';
            let result_mata_pelajaran = await
new Promise((resolve, reject) => {

connection.query(query_mata_pelajaran,
[list_kode_mapel[i]], (error, results) => {
                    if (error) {
                        reject(error);
                    } else {
                        resolve(results);
                    }
                })
            })
            //
console.log(result_mata_pelajaran[0].mat
pel)
            list_mata_pelajaran[i] =
result_mata_pelajaran[0].matpel
        }
        console.log(list_mata_pelajaran)

        return res.status(200).json({ success:
true, data: { mata_pelajaran_counter,
list_mata_pelajaran }, status: true });
    } catch (error) {
        console.log(error, "error
menghitung mata pelajaran");
        return res.status(500).json({ success:
true, message: error.message, status:
true });
    }

})

router.get('/data/:nip', async (req, res)
=> {
    const query = 'SELECT * FROM
tb_guru WHERE nip = ?';
    const { nip } = req.params;

    try {
        const result = await new
Promise((resolve, reject) => {
            connection.query(query, [nip],
(error, results) => {
                if (error) {
                    reject(error);
                } else {
                    resolve(results);
                }
            })
        })

        return res.status(200).json({ success:
true, data: result[0], status: true });
    } catch (error) {
        console.log(error, "error mengambil
data guru");
        return res.status(500).json({ success:
true, message: error.message, status:
true });
    }
})


router.get('/wali-kelas', async (req, res)
=> {
```

```
    res.sendFile(__dirname +
'/ui/guru/wali_kelas.html');
})

router.get('/wali-kelas/get/:nip', async
(req, res) => {

    const { nip } = req.params;
    console.log(nip)
    const query = 'Select * from tb_kelas
where nip_guru = ?';
    try{
        const result = await new
Promise((resolve, reject) => {
            connection.query(query, [nip],
(error, results) => {
                if (error) {
                    reject(error);
                } else {
                    resolve(results);
                }
            })
        })
        console.log(result)
        // console.log(result[0].id_kelas)
        if (result.length == 0) return
res.status(200).json({ success: true, data:
[], status: true });

        const query_siswa = 'SELECT * from
tb_siswa where nama_kelas = ?';

        const result_siswa = await new
Promise((resolve, reject) => {
            connection.query(query_siswa,
[result[0].id_kelas], (error, results) => {
                if (error) {
                    reject(error);
                } else {
                    resolve(results);
                }
            })
        })
        return res.status(200).json({ success:
true, data: result_siswa, nama_kelas:
result[0].nama_kelas, status: true });
    } catch (error) {
        console.log(error, "error mengambil
data guru");
        return res.status(500).json({ success:
true, message: error.message, status:
true });
    }
})


module.exports = router
```

4. ortu_route.js

```
const express = require('express');
const router = express.Router();
const dotenv = require('dotenv');
const conn = require('../conn');
const connection = conn.connection;
const path = require('path');

dotenv.config();

router.get('/', async (req, res) => {
    res.sendFile(__dirname +
'/ui/ortu/index.html');
})

router.get('/absensi', async (req, res) =>
{
    res.sendFile(__dirname +
'/ui/ortu/absensi.html');
})

router.get('/gambar/:gambar', async
(req, res) => {
    const { gambar } = req.params;
    // console.log(gambar)
    const dir = path.join(__dirname,
'../image/' + gambar);
    // console.log(dir)
    res.sendFile(dir);
})

router.get('/data/:nis', async (req, res) =>
{
    const nis = req.params.nis
```

```
    const query = 'SELECT
a.nis,a.nama,a.kelas,a.angkatan,a.foto,b.i
d_kelas,c.nik,c.tempat_lahir,c.tanggal_lahi
r,c.jenis_kelamin,c.agama,c.orang_tua_1,c
.orang_tua_2,c.no_hp_orang_tua,c.alamat
FROM tb_siswa a join tb_kelas b join
tb_detail_siswa c on
a.nama_kelas=b.id_kelas and a.nis=c.nis
where a.nis = ?';
    try {
        const result = await new
Promise((resolve, reject) => {
            connection.query(query, [nis],
(error, results) => {
                if (error) {
                    reject(error);
                } else {
                    resolve(results);
                }
            })
        })
        return res.status(200).json({ success:
true, data: result[0], status: true });
    } catch (error) {
        console.log(error);
        return res.status(500).json({ success:
true, message: error.message, status:
true });
    }
})


module.exports = router
```

KARTU MONITORING BIMBINGAN

MAHASISWA PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS MUHAMMADIYAH PAREPARE

PROPOSAL

Mahasiswa : NURUL ANNISA A	Pembimbing I : ADE HASTUTY S. Kom., ST., MT
NIM : 217280184	Pembimbing II : WAHYUDDIN S. Kom.,M. Kom
Judul Skripsi : Produk Media Pembelajaran Digital AR Berbasis Android IPA Sistem Tata Surya SMP 5 Dua Pitue SIDRAP	

ARAHAN PEMBIMBING I	HARI/TGL & PARAF PEMBIMBING	ARAHAN PEMBIMBING II	HARI/TGL & PARAF PEMBIMBING
Konsultasi 1		Konsultasi 1	
Konsultasi 2		Konsultasi 2	
Konsultasi 3		Konsultasi 3	
Konsultasi 4		Konsultasi 4	
Konsultasi 5		Konsultasi 5	

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

KARTU MONITORING BIMBINGAN

MAHASISWA PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS MUHAMMADIYAH PAREPARE

SKRIPSI
~~PROPOSAL~~

Mahasiswa : NURUL ANNISA A	**Pembimbing I : Ade Hastuty, S.Kom.,ST.,MT.**
NIM : 217280184	**Pembimbing II :Wahyuddin, S.Kom., M.Kom.**
Judul Skripsi : SISTEM INFORMASI ADMINISTRASI AKADEMIK SMP NEGERI 5 DUA PITUE SIDRAP	

ARAHAN PEMBIMBING I	HARI/TGL & PARAF PEMBIMBING	ARAHAN PEMBIMBING II	HARI/TGL & PARAF PEMBIMBING
Konsultasi 1 Isi data real, dimulai dari data bapa NIP, dan surat keluar		Konsultasi 1 - Tampilkan actor siswa pada aplikasi	
Konsultasi 2 halaman utama disusuaun sesuai dengan sistem rancangan yg di usulkan		Konsultasi 2 - Tampilkan Actor wali kelas pada aplikasi	
Konsultasi 3 harus ada font untuk cetak atau print		Konsultasi 3 - Masukkan kamus data pada dokumen	
Konsultasi 4		Konsultasi 4 Isi	
Konsultasi 5		Konsultasi 5	

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