임베디드시스템 프로그래밍II

프로젝트명	QR 출결 관리 시스템
순번	23
학번	202012674
이름	홍 서 정

2023 - 06 -19

00 목차

01 프로젝트 주제

모바일 학생증의 QR코드를 활용한 출석 체크 시스템

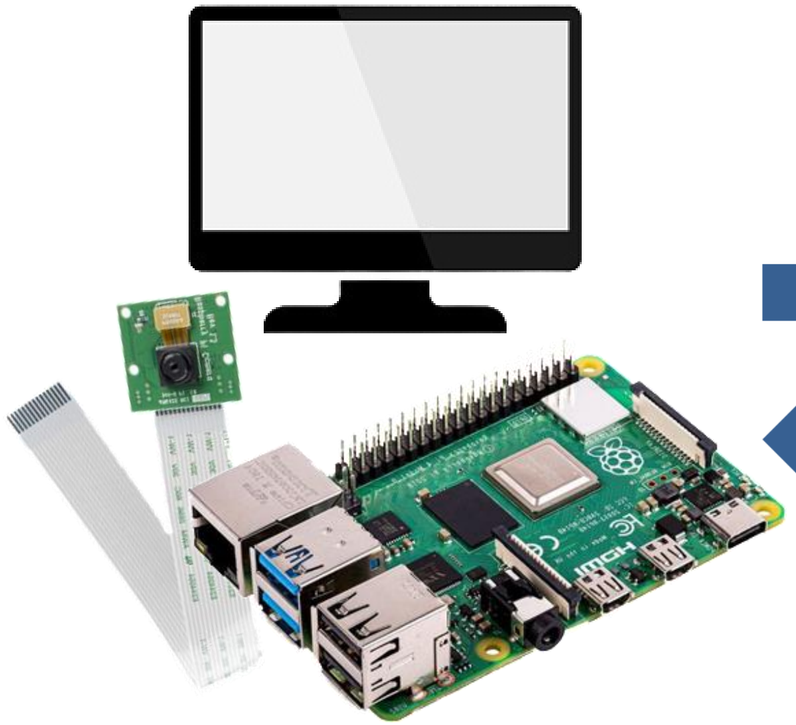
01 프로젝트 주제

PK 02020126742023061210142826002

PK	202012674	20230612	101428
- 학교 코드 - 올바른 QR인지 판단	- 학번 - 수강 정보 확인 및 출결 데이터 저장	- 날짜 - 출석 시간 확인	- 시간 - 출석 시간 확인

02 | 사용 기술

입력



라즈베리파이

처리

결과

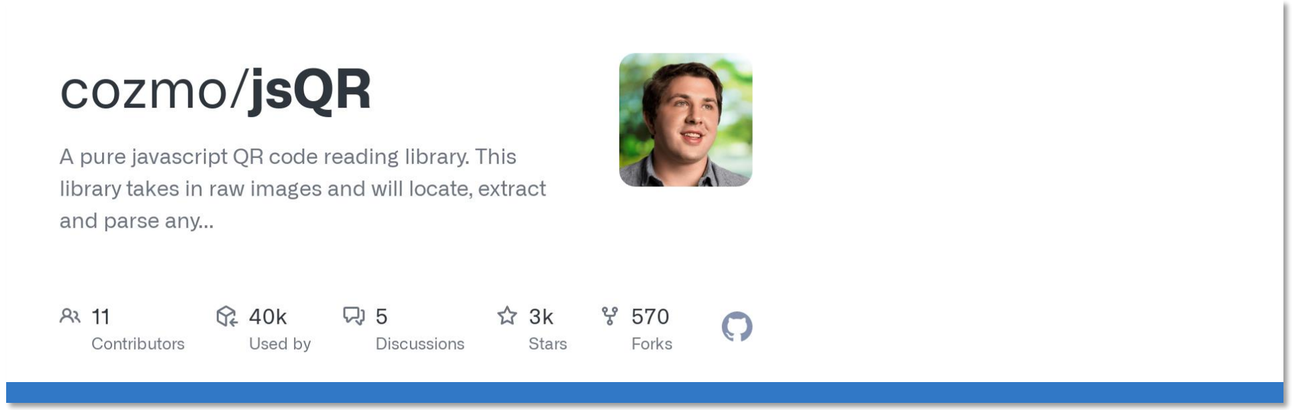
오픈소스 jsQR

JS

node

MySQL

02 | 사용 기술 – 오픈소스 jsQR



- 자바스크립트로 QR 코드를 읽는 라이브러리
- 카메라로부터 원시 이미지를 가져와 QR 코드를 인식
- 디코딩된 QR코드 데이터 제공

03 결과물 – 강의 검색 화면

QR체크 화면

라즈베리파이 카메라로
QR 코드 스캔

오픈소스로 QR코드를 인식

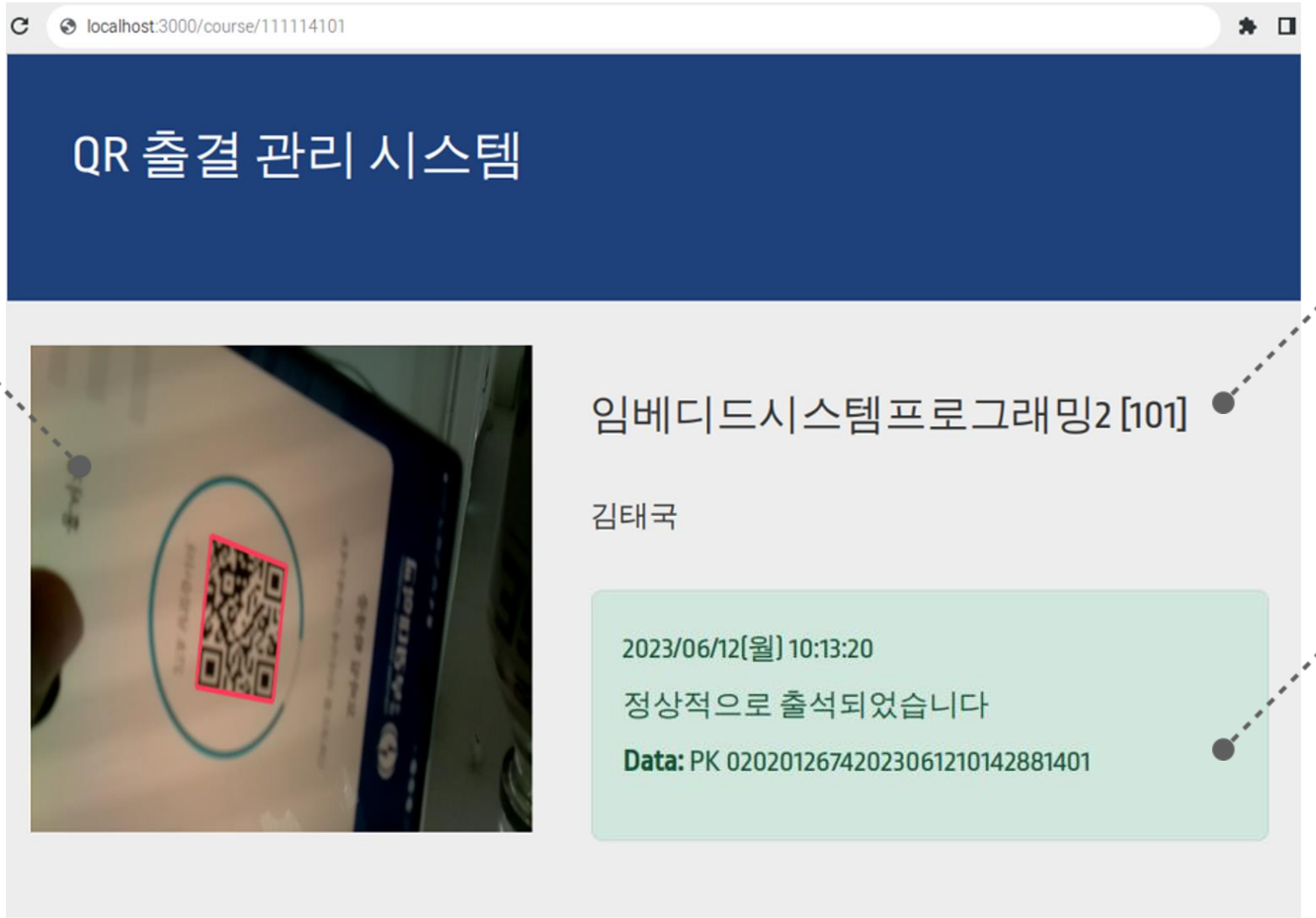
학번으로 강의 검색

QR로 읽어들인 학번 정보

수강 목록

원하는 강의 페이지로 이동

03 결과물 – 출석체크 화면



QR체크 화면

라즈베리파이 카메라로
QR 코드 스캔

오픈소스로 QR코드를 인식

강의 정보 표시

학수번호에 따라
강의 정보 표시

출결 안내 메세지

출석 결과 메세지를 출력

04 코드 설명 – QR인식 시

```
//QR코드 인식 여부 판단
if (code) {
  //인식한 문자열을 구조체로 저장
  let str = code.data;

  var qrdata = {
    check: str.slice(0, 2),       //올바른 QR인지 판단
    hakbun: str.slice(8, 17),     //학번
    date_y: str.slice(17, 21),    //날짜 - 연
    date_m: str.slice(21, 23),    //날짜 - 월
    date_d: str.slice(23, 25),    //날짜 - 일
    time_h: str.slice(25, 27),    //시간 - 시
    time_m: str.slice(27, 29),    //시간 - 분
    time_s: str.slice(29, 31)     //시간 - 초
  }
  ...
}
```

QR코드에 저장된 데이터를 분할해 구조체로 저장

올바른 QR 형식인지 판단

```
if (qrdata.check == 'PK') {                  //올바른 QR인지 판단
  if (currentPath.startsWith('/course')) {   //출석 페이지인 경우
    axios.post('/insertData', { qrdata })    //서버로 qr데이터 전송(출석)
      .then(function (response) {
        console.log(response.data);
    })
    ...
    //메인페이지 또는 검색페이지인 경우
  } else if (currentPath.startsWith('/search') || currentPath == '/') {
    location.href = "/search/" + qrdata.hakbun; //검색 결과 페이지로 이동
  }
} else {                                     //잘못된 QR인 경우
  console.log('QR_ERR');
}
```

1. 강의 검색 페이지 이동
   /search/qrdata.hakbun

2. 서버로 QR데이터 전송
   post('/insertData', {qrdata})

2023/06/12(월) 10:38:31

잘못된 QR입니다

Data: http://www.Online-QRCode-Generator.com

04 코드 설명 – 강의 검색

/search/202012674

```
app.get('/search/:hakbun', function (req, res) {
 //URL 경로에서 학번 추출
 hakbun = req.params.hakbun;

 //수강 목록을 검색하기 위한 쿼리
 const query = 'SELECT * FROM COURSE NATURAL JOIN ENROL WHERE S_NUM = ?';

 //쿼리 결과를 수강목록 페이지로 전달
 connection.query(query, [hakbun], (err, results) => {
    ...
   res.render('courselist.ejs', { results: results }, function (err,html){
      res.send(html)
    })
  })
});
```

```
SELECT * FROM COURSE NATURAL JOIN ENROL WHERE S_NUM='202012674'
```

c_num	c_name	professor	s_num
103932102	운영체제 [102]	김창수	202012674
111112102	인터넷DB응용 [102]	박선이	202012674
111114101	임베디드시스템프로그래밍2 [101]	김태국	202012674

```
<tr>
  <th> <%= results[i].c_num %> </th>
  <td> <%= results[i].c_name %> </td>
  <td> <%= results[i].professor %> </td>
  <td> <button onclick="location.href='/course/<%= results[i].c_num %>'"> 이동 </button></td>
</tr>
```

학수번호	교과목명	담당교수	
103932102	운영체제 [102]	김창수	이동
111112102	인터넷DB응용 [102]	박선이	이동
111114101	임베디드시스템프로그래밍2 [101]	김태국	이동

/course/학수번호 하이퍼링크 생성

04 코드 설명 – 출석체크(1)

/course/111114101

```
app.get('/course/:id', async (req, res) =>
{  //URL경로에서 학수번호 추출
  courseid = req.params.id;
```

```
SELECT * FROM COURSE_TIME
```

	c_num	day	start_time	end_time
▶	103932102	월	14:00:00	15:50:00
	103932102	화	17:00:00	17:50:00
	111114101	월	09:00:00	12:50:00

전달받은 QR데이터의 학번으로
수강 여부 및 강의 시간 검사

```
app.post('/insertData', async (req, res) => {

  let qrdata = req.body.qrdata;
  let hakbun = qrdata.hakbun;

  ...

  try {
    //1. 수강 여부 검사
    const q1 = await connection.promise().query(`SELECT * FROM ENROL
                                                  WHERE C_NUM=? AND S_NUM=?;`, [courseid, hakbun]);

    //해당 과목에 등록된 경우
    if (q1[0].length > 0) {

      //2. 강의 시간 검사 – 해당 요일에 수업이 있는지
      const q2 = await connection.promise().query(`SELECT * FROM COURSE_TIME
                                                    WHERE C_NUM=? AND DAY=?`, [courseid, week[day]]);
      if (q2[0].length > 0) {
        //해당 요일에 수업이 있는 경우

        ...

      } else {
        //해당 요일에 수업이 없는 경우
        status = "CLOSED";
        console.log(`해당 과목의 출석 시간이 아닙니다`);
      }

    } else {
      //해당 과목에 등록되지 않은 경우
      status = "NOT_ENROL";
      console.log(`해당 과목을 수강하지 않습니다`);
    }

  }
});
```

```
SELECT * FROM ENROL
```

	c_num	s_num
▶	103932102	202012674
	111112102	202012674
	111114101	202012674

2023/06/12(월) 10:39:01

해당 과목의 출석 시간이 아닙니다

Data: PK 02020126742023061210402431300

2023/06/12(월) 10:14:06

해당 과목에 등록되지 않은 QR입니다

Data: PK 02020126742023061210153015480

04 코드 설명 – 출석체크(2)

```
SELECT * FROM COURSE_TIME
```

	c_num	day	start_time	end_time
▶	103932102	월	14:00:00	15:50:00
	103932102	화	17:00:00	17:50:00
	111114101	월	09:00:00	12:50:00

수업시간 데이터를 사용해
출석 인정 시간 설정

```
//2. 강의 시간 검사 – 해당 요일에 수업이 있는지
const q2 = await connection.promise().query(`SELECT * FROM COURSE_TIME
                                              WHERE C_NUM=? AND DAY=?`, [courseid, week[day]]);
//해당 요일에 수업이 있는 경우
if (q2[0].length > 0) {

  const start_time = q2[0].start_time;                //강의 시작 시간
  const end_time = q2[0].end_time;                    //강의 종료 시간
  const qrdate = new Date(`${qr_date} ${qr_time}`);   //출석 기준 시간

  //출석 인정 시간 설정
  //출석 시작 시간 : 수업 시작 10분 전부터 출석 시작
  var set_presenttime = new Date(`${current_date} ${start_time}`);
  set_presenttime.setMinutes(set_presenttime.getMinutes() - 10);

  //지각 시작 시간 : 수업 시작 후 10분까지 정상 출석
  var set_latetime = new Date(`${current_date} ${start_time}`);
  set_latetime.setMinutes(set_latetime.getMinutes() + 10);

  //출석 종료 시간 : 수업 종료 30분 전까지 지각 인정
  var set_endtime = new Date(`${current_date} ${end_time}`);
  set_endtime.setMinutes(set_endtime.getMinutes() - 30);
```

QR코드의 시간정보와 비교 후
해당되는 상태 값 저장

```
if (set_presenttime <= qrdate && qrdate < set_latetime) {
    status = 'PRESENT';
    console.log(`정상출석`);
 } else if (set_latetime <= qrdate && qrdate < set_endtime) {
    status = "LATE";
    console.log(`지각`);
 } else {
    status = "CLOSED";
    console.log(`해당 과목의 출석 시간이 아닙니다`);
 }
```

2023/06/12(월) 10:13:20

정상적으로 출석되었습니다

Data: PK 020201267420230612101428814O1

2023/06/12(월) 10:12:08

지각입니다

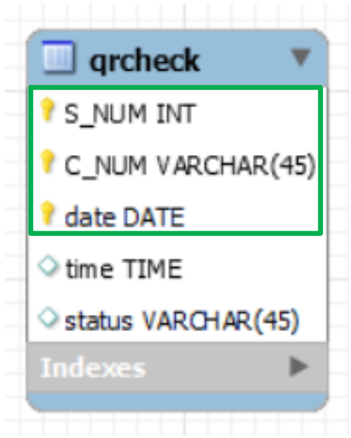
Data: PK 020201267420230612101327740 47

2023/06/12(월) 10:39:01

해당 과목의 출석 시간이 아닙니다

Data: PK 020201267420230612104024313 00

04 코드 설명 – 출석체크(3)



중복 출석 시
기본 키 중복 오류 발생

```
//출결 데이터 저장 쿼리(학번 – 과목코드 – 출석날짜 – 출석시간 – 출석상태)
const insertQuery = `INSERT INTO QRCHECK (s_num, c_num, date, time, status)
                                  VALUES (?, ?, ?, ?, ?)`;
params = [hakbun, courseid, current_date, current_time, status];


  if (status =='PRESENT' ||status =='LATE' ){
    //출석 데이터 삽입
    const results = await connection.promise().query(insertQuery, params);
  }
  console.log('데이터 삽입 성공');

  ...

} catch (error) {
  //데이터 삽입 오류 – 중복
  console.error('오류:', error);
  status = 'DUPLICATE';
}
```

```
SELECT * FROM QRCHECK
```

S_NUM	C_NUM	date	time	status
202012674	103932102	2023-06-14	15:41:42	LATE
202012674	111114101	2023-06-14	15:40:38	PRESENT

정상출석 또는 지각인 경우
출결 데이터 삽입

2023/06/12[월] 10:12:14

이미 출석된 QR입니다

Data: PK 0202012674202306121013277 4047

05 시연