

캡스톤설계프로젝트

Kingo Manager

Team AIU

1. Frontend

2. Backend

3. AI

1. Frontend - UI & UX : 회원가입

The image displays three sequential mobile app screens for a sign-up process. Each screen has a status bar at the top showing the time 3:23, a black pill-shaped home indicator, and signal/battery icons. A back arrow is visible in the top-left corner of each screen.

Screen 1: Select your grade (1/3)
Title: 본인의 학년을 선택해 주십시오 (1/3)
List of options:

- ☐ 1학년
- ☐ 2학년
- ☐ 3학년
- ☐ 4학년 이상

Next button: Next

Screen 2: Check for existing items (2/3)
Title: 이미 획득한 상품이 있다면 체크해 주십시오 (2/3)
List of items with checkboxes:

- ☒ 민생품
- ☒ 장의품
- ☐ 국제품

Next button: Next

Screen 3: Select your interests (3/3)
Title: 본인의 관심사를 선택해 주십시오 (3/3)
List of interests with checkboxes:

- ☐ 학점관리
- ☐ 자기이해와 진로탐색
- ☒ 봉사
- ☐ 동아리, 학회
- ☐ 외국어 필수 획득(국제품)
- ☐ 자격증
- ☐ 공모전/외부활동
- ☐ 졸업요건
- ☐ 기업인턴십/산학협력
- ☐ 취업캠프/서류, 면접 클러닉
- ☐ 대학원/학석연계

Sign up button: Sign up

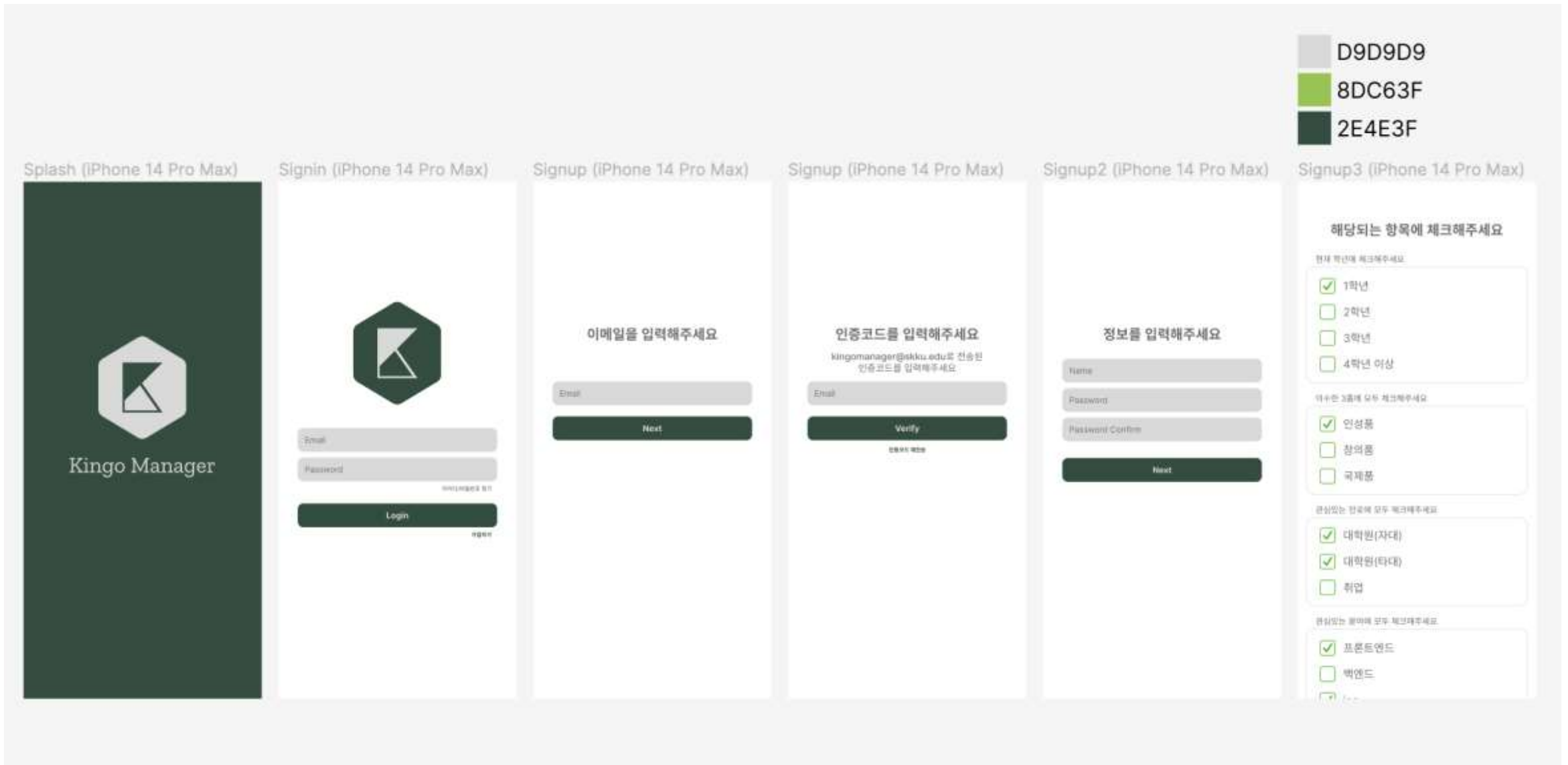
A toast message at the bottom left of the first screen reads: AsyncStorage has been extracted from react-na...

1. Frontend - UI & UX : 회원가입

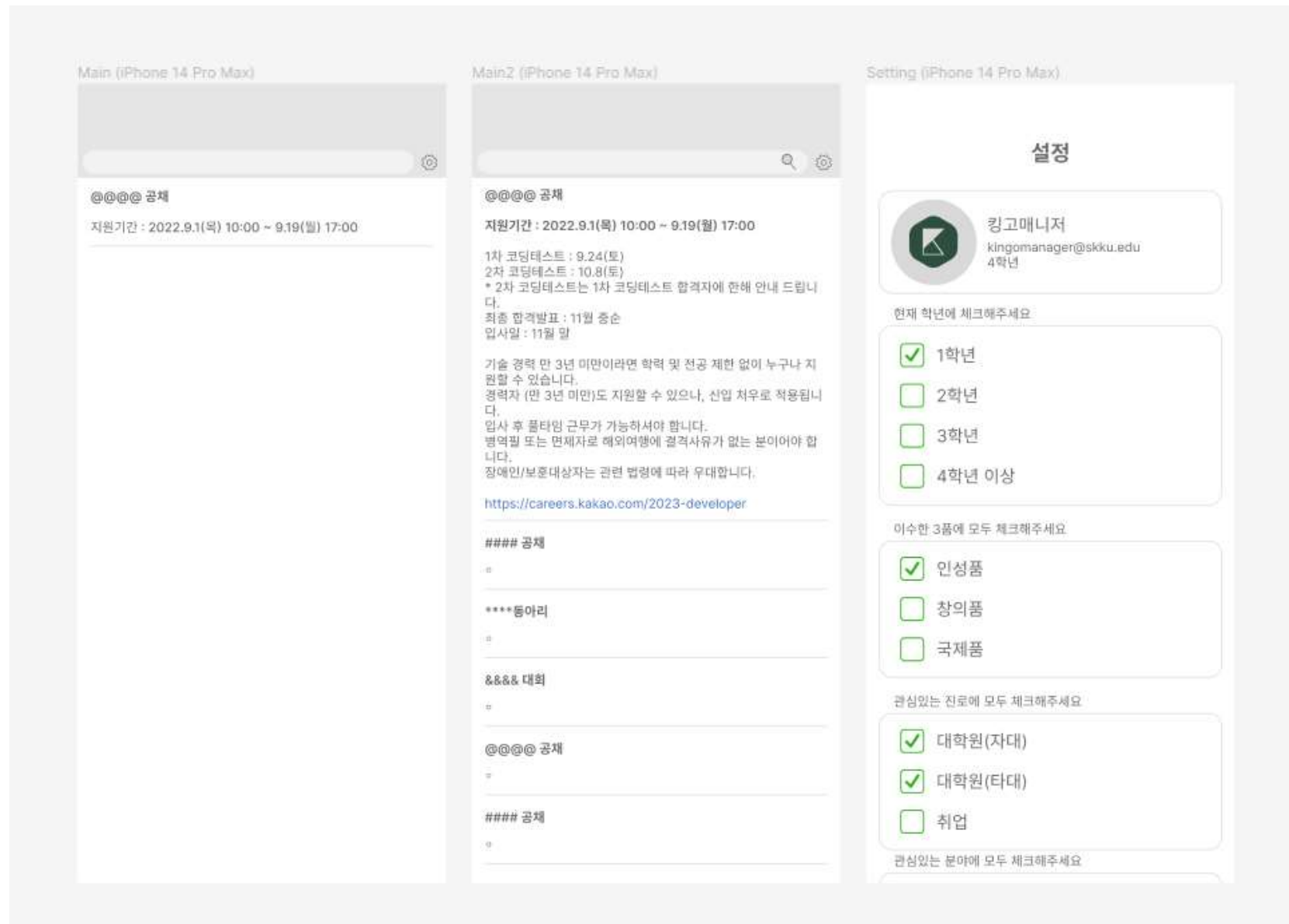
```
const _handleSignupButtonPress = async (event) => {
  setLoading(true);
  try {
    const response = await axios.post(`${baseUrl}/adduser`, {
      name: name,
      email: email,
      password: password,
      grade: checkgrade,
      insung: insung,
      changeui: changeui,
      gukje: gukje,
      interest: interest,
    });
    Alert.alert('Success');
    navigation.navigate('Signin');
  } catch (error) {
    if (error.response) {
      const errorResponse = error.response;
      console.log('data : ', errorResponse.data);
      console.log('status : ', errorResponse.status);
      console.log('headers : ', errorResponse.headers);
    } else if (error.request) {
      console.log('request : ', error.request);
    } else {
      console.log('meessage : ', error.message);
    }
  } finally {
    setLoading(false);
  }
};
```

1. 로그인/로그아웃을 처음에 firebase
로 구현하였다가 axios를 이용한
http통신으로 변경하였습니다.

1. Frontend - UI & UX : 회원가입



1. Fontend - UI & UX : 메인화면



현재 아직 구현하지 못한 부분

1. Async-storage를 이용한 게시물의 로컬 저장

2. 가입시 이메일을 통한 인증

Trial & error

1.지금까지 받은 추천 목록의 저장법에 대해 고민

- (1) 로그인할 때마다 유저에 대한 데이터를 전부 불러온다
-> 추후 데이터의 양이 쌓이면 로그인시 시간이 상당히 소요된다. (x)
- (2) 카카오톡처럼 거의 로그인할 일이 없기 때문에 토큰을 이용하여 로그인 상태를 유지하고 데이터는 로컬에 저장한다. (o)

2. 가입시 이메일을 통한 인증

Table Column redefine

```
# define Table : class User, users
class User(Base):
    __tablename__ = "users"
    id = Column(Integer, primary_key=True)
    email = Column(String(50), unique=True)
    password = Column(String(50), unique=False)
    name = Column(String(50), unique=False)
    grade = Column(Integer, unique=False)
    insung = Column(Boolean, unique=False)
    changeui = Column(Boolean, unique=False)
    gukje = Column(Boolean, unique=False)
    interest = Column(Integer, unique=False)
```

```
# define Table : class Recommend, recommends
class Recommend(Base):
    __tablename__ = "recommends"
    id = Column(Integer, primary_key=True)
    email = Column(String(50), unique=False)
    title = Column(String(100), unique=False)
    category = Column(String(100), unique=False)
    contents = Column(String(500), unique=False)
    link = Column(String(100), unique=False)
```

Encoding issue

성균관대학교
공지사항



encoding

Utf-8

%EC%84%B1%EA%B7
%A0%EA%B4%80%EB
%8C%80%ED%95%99
%EA%B5%90%20%EA
%B3%B5%EC%A7%80
%EC%82%AC%ED%95
%AD



decoding

성균관대학교
공지사항

Model to Post

```
> __pycache__
  five_content.py
  layers.py
  main.py
  param_parser.py
  post.py
  Predict_model.py
  predict.py
  test.py
  trainer_and_networks.py
  utils.py
  readme.md
```

```
def Choose_five():
    if(main() == 1):
```

```
    return json_list
```

```
def Post(input):
    for i in input:
        headers = {'Content-Type': 'application/json',}

        #json_data = input
        json_data = i

        url = 'https://31lwwh4zd4.execute-api.ap-northeast-1.amazonaws.com/prod/lambda-test'

        response = requests.post(url=url, headers=headers, data=json_data)

        #print(response.text)
        #print(response.status_code)
```

Thank you!