รายงานความก้าวหน้าวิชา CE Project 2

ครั้งที่ 3

ระหว่างวันที่ 04 ก.พ. 66 ถึงวันที่ 23 ก.พ. 66

1. ชื่อโครงงาน (อังกฤษ) Curriculum Output Prediction From Student Academic Data
2. การดำเนินงานมีความก้าวหน้า 65 % (ใช้ค่า **% Complete** จาก MS Project)

   มีความก้าวหน้าเพิ่มขึ้นจากรายงานความก้าวหน้า ครั้งก่อน 5 %

   ☐ เร็วกว่าแผน ..... วัน ☐ ช้ากว่าแผน .....49..... วัน

3. รายละเอียดความก้าวหน้า

นัดประชุมกับที่ปรึกษาจำนวน 1 ครั้ง

ครั้งที่ 1 : การประชุมในครั้งที่ผ่านมาได้มีหัวข้อการประชุมดังนี้ 1) พูดคุยถึง requirement progress 2) แนวทางในการพัฒนาต่อ

1) พูดคุยถึง requirement progress ทางผู้จัดทำได้อธิบายแนวทางการพัฒนาในปัจจุบันให้ทางที่ปรึกษาโดยมีรายละเอียดดังนี้ โดยปัจจุบันงานจะเน้นการทำพัฒนาในส่วนของหน้าการใช้งานของ web application หรือ frontend โดยทางการพัฒนาในส่วนการทำงานหลังบ้านหรือ backend นั้นจะเป็นการปรับแต่ง algorithm ของการทำงานให้ง่ายต่อการ scalability มากขึ้นและขนานไปกับตัว design ของ frontend
2) แนวทางการพัฒนาต่อนั้นจากที่ได้พูดคุยกับทางที่ปรึกษานั้น ทางที่ปรึกษาได้แนะแนวให้ทางผู้จัดทำนั้นดังนี้โดยได้มีการพูดคุยกับที่ปรึกษาเรื่องการปรับเปลี่ยน schema ของ model โดยการเปลี่ยน schema ครั้งนี้ทำเพื่อลดความซ้ำซ้อนของข้อมูลในระบบและจัดรูปแบบของ schema เพื่อให้การ query ข้อมูลไป process นั้นสะดวกยิ่งขึ้นโดยจะทำการปรับ schema ของ table student โดยจะทำการ normalize ออกเป็น student_data และ student_grade

หัวข้อการพัฒนาโครงงานตาม Gantt Chart

ศึกษาทฤษฎีที่เกี่ยวข้อง Complete 100 % หมายเหตุ ในส่วนการพัฒนาในปัจจุบันนั้นไม่ทฤษฎีที่ต้องศึกษาเพิ่มเติม

เตรียม Data สำหรับการพัฒนา Complete 100 % หมายเหตุ เนื่องด้วยในช่วง progress ที่ผ่านมาทางที่ปรึกษาได้ทำการนำข้อมูลที่ทางผู้จัดทำได้รวบรวมมาจากการทำแบบสอบถามการมีงานทำของบัณฑิตมาทำการมาเข้ารหัสข้อมูลรหัสนักศึกษาหลังจากนั้นทางผู้จัดทำได้นำข้อมูลในส่วนนั้นมาเพิ่มลงใน column ของ student_data

พัฒนา Job Classification Model Complete 95 % หมายเหตุเนื่องด้วยทางผู้จัดทำเพิ่งได้รับมอบข้อมูลส่วนที่เข้ารหัสมาจึงยังจำเป็นต้องทำการทดลองกับข้อมูลเพื่อหาการจัดกลุ่มของหมวดหมู่อาชีพเพื่อลดความผิดพลาดของการทำนายอาชีพของ model jobclassification

พัฒนา Django Web App Complete 36% (remaining 165.75 Hr) หลังจากที่ได้ทำการคุยแนวทางการพัฒนาต่อกับที่ปรึกษาในครั้งก่อนนั้นได้ทำการ optimize algorithms ในการทำงานของ function ทั้งหมดใน application backend และ recommend ทั้งหมด โดยการ optimize ส่วนใหญ่จะทำการเน้นไปยังในด้านการ scalability เพื่อให้ทุก api แล้วทุก logic function ที่ทำการอยู่ในส่วนหลังบ้านของ web application นั้นพร้อมสำหรับรองรับการทำงานในทุกๆรูปแบบ หรือพร้อมนำไปใช้งานได้ในทุกๆหลักสูตรการศึกษา ตัวอย่างของการ optimize ดังรูปที่ 1 คือ function generateModel version ก่อนที่จะทำการ optimize โดยจะมีข้อจำกัดอยู่หลายส่วนเช่นตัว function จะ query แค่ข้อมูลจาก หลักสูตรวิศวกรรมคอมพิวเตอร์ และ วิศวกรรมคอมพิวเตอร์(ต่อเนื่อง) และมีส่วนของการทำงานที่ไม่จำเป็นซึ่งอาจทำให้ระบบเกิดข้อขวดในระบบโดยไม่จำเป็นขึ้นมา โดยการ optimize .ในส่วนของ function นี้นั้นจะเน้นไปทำการปรับลดการขั้นตอนการทำงานที่ไม่จำเป็นออก และ ปรับให้ function รองรับ body โดยใน body จะเก็บข้อมูลที่จำเป็นในการสร้าง model ขึ้นมา ดังรูปที่ 2

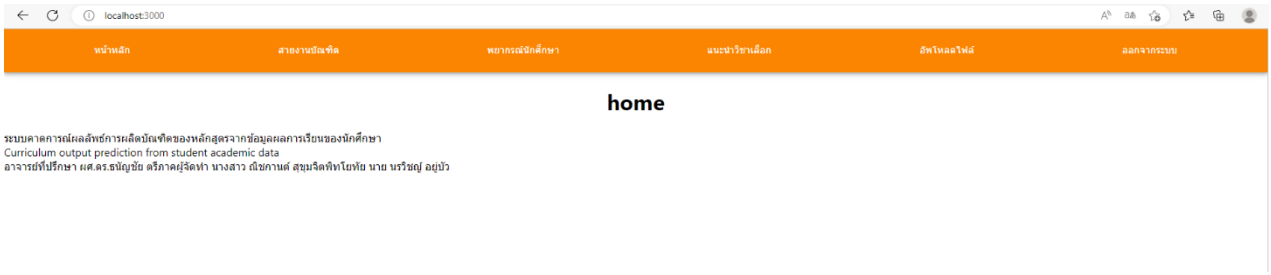
```
578
579 @csrf_exempt
580 def generateModel(request, curri):
581     if request.method == 'POST':
582       todays_date = date.today()
583       this_year = todays_date.year + 543 - 4
584       base_year = 2560
585       year_that_grad = []
586       query = Q(start_year = str(base_year))
587       for runnub in range(base_year+1, this_year+1):
588         query.add(Q(start_year = str(runnub)), Q.OR)
589       body_unicode = request.body
590       body = json.loads(body_unicode)
591       model_name = body['name']
592       model_pred = body['pred']
593       print(model_pred)
594       qdata = list(Student.objects.filter(query).values())
595       q_subject_data = list(Subject_Data.objects.all().values())
596       df_subject = pd.DataFrame(q_subject_data)
597       thisdict = genSubjectDict(df_subject)
598       df = pd.DataFrame(qdata)
599       dfs = queryBycurriculum(df)
600       dfs = transfromAlldfs(dfs)
601       for i in dfs:
602         temp = dfs[i][0]
603         temp = temp[temp.grade != 'Zero']
604         dfs[i].append(temp)
605       # q_subjectClassAndsubId = 'SELECT subject_id, subject_class from df_subject;'
606       # df_subject = sqldf(q_subjectClassAndsubId)
607       if model_pred == 'Class':
608         for j in dfs:
609           dfs[j][0]['subject_class'] = dfs[j][0].apply (lambda row: findSubjectClass(row['subject_id'], thisdict), axis=1)
610           dfs[j][1]['subject_class'] = dfs[j][1].apply (lambda row: findSubjectClass(row['subject_id'], thisdict), axis=1)
611           # casttemp = dfs[j][1]
612           # casttemp[["grade"]] = casttemp[["grade"]].apply(pd.to_numeric)
613           # dfs[j][1] = casttemp
614         for k in dfs:
615           tempAVG = dfs[k][1]
616           q_find_AVG = "SELECT student_id, AVG(grade) as grade, semester, year, curriculum, subject_class FROM tempAVG GROUP BY subject_class, student_id ORDER BY student_id"
617           tempDFS_AVG = sqldf(q_find_AVG)
618           dfs[k][1] = tempDFS_AVG
619       min_rating = 0
620       max_rating = 4
```

รูปที่ 1 function generateModel ตัวเก่า

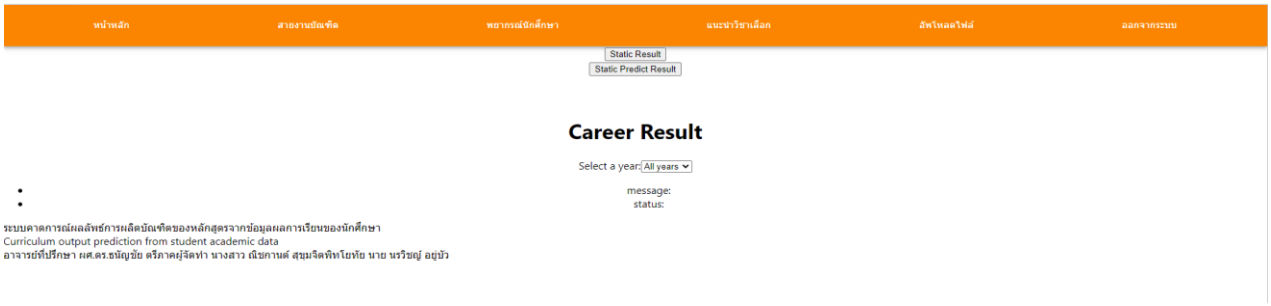
```python
@csrf_exempt
def generate_rec_model(request):
  if request.method == 'POST':
    if request.body:
      body = json.loads(request.body)
      model_name = body['name']
      model_year = body['year']
      model_type = body['type']
      model_curri = body['curriculum']
      curri_year = int(model_year)
      while curri_year % 4 != 0:
        curri_year = curri_year - 1
      curri_year = str(curri_year)
      s_data = pd.DataFrame(list(Student_Data.objects.all().values()))
      s_grade = pd.DataFrame(list(Student_Grade.objects.all().values()))
      sub_data = list(Subject_Data.objects.filter(year = curri_year).values())
      sub_data_df = pd.DataFrame(sub_data)
      thisdict = genSubjectDict(sub_data_df)
      join_q = "select s_grade.student_id, s_grade.subject_id, s_grade.grade, s_data.career, s_data.curriculum, s_data.status, s_data.curriculum_year from s_grade left join s_data on s
      train_data = sqldf(join_q)
      train_data = train_data.loc[(train_data['grade'] != 'Zero') & (train_data['grade'] != 'nan') & (train_data['curriculum'] == model_curri) & (train_data['status'] == 'graduate') &
      train_data = transfromGrade(train_data)
      if model_type == 'class':
        train_data['subject_class'] = train_data.apply (lambda row: findSubjectClass(row['subject_id'], thisdict), axis=1)
        group_query = "select student_id, subject_class, round(avg(grade), 2) as grade from train_data group by subject_class, student_id order by student_id"
        train_data = sqldf(group_query)
        return_model = train_rec_model(train_data)
      else:
        return_model = train_rec_model(train_data,1)
      this_rmse = str(return_model['rmse'])
      rec = SurpriseModel(name = model_name, curriculum = model_curri, rmse = this_rmse, type_pred = model_type, rec_model = return_model['model'])
      rec.save()
      res = {"message": f'Model successfuly create name : {model_name} for : {model_curri} type : {model_type} with rmse : {this_rmse}.', "status" : status.HTTP_200_OK}
    else:
      res = {"message": "Pls select model type and curriculum", "status": status.HTTP_400_BAD_REQUEST}
  else:
    res = {"message": "Method not match.", "status": status.HTTP_400_BAD_REQUEST}
  return JsonResponse(res , safe=False, json_dumps_params={'ensure_ascii': False})
```

รูปที่ 2 function generateModel ตัวเก่า

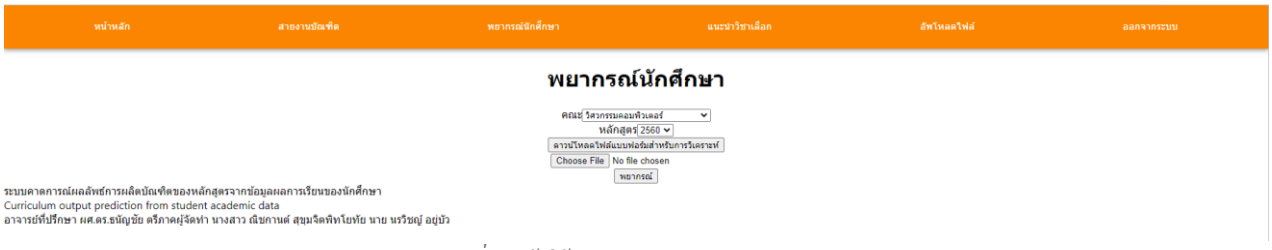
พัฒนา Frontend Complete 42% (remaining 22.33 Hr) การพัฒนาของ frontend นั้นจะมีความก้าวหน้าตามรูปภาพ



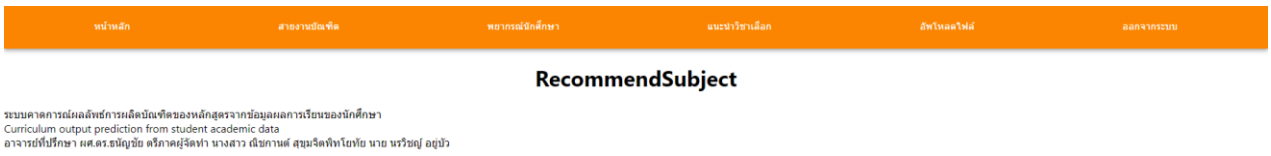
รูปที่ 3 หน้าหลักของ web application



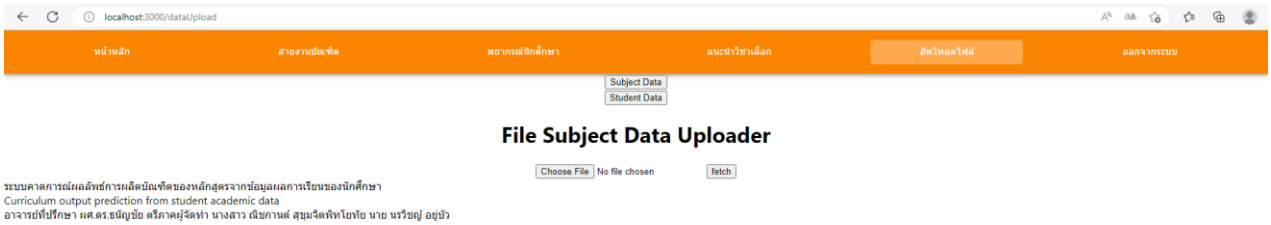
รูปที่ 4 หน้าแสดงจำนวนอาชีพทั้งหมด



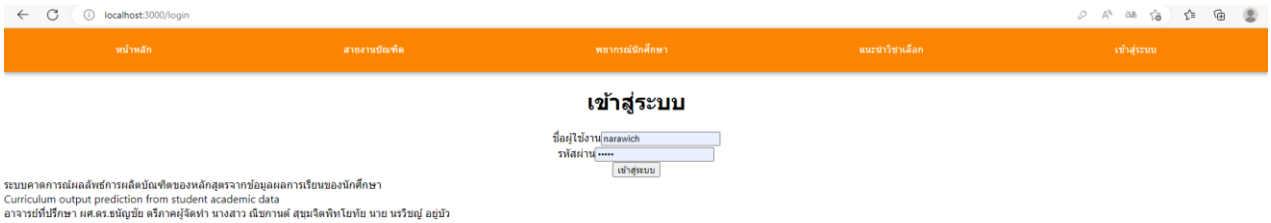
รูปที่ 5 หน้าใช้งานระบบการทำนายอาชีพ



รูปที่ 6 หน้าใช้งานระบบการแนะนำรายวิชา



รูปที่ 7 หน้าใช้งาน upload file



รูปที่ 8 หน้าใช้งานระบบ login

4. ปัญหาที่เกิดขึ้นและแนวทางการแก้ไข

ปัญหาจากครั้งก่อน

1. ปัญหาการ prediction รายวิชาที่ไม่ครบ เนื่องด้วยการ optimize ที่กล่าวไปข้างต้นนั้นทำให้ทางผู้จัดทำได้ทำการสำรวจการทำงานของ algorithms ของ function ใหม่อีกครั้งแล้วพบว่า function นั้นไม่ได้คืนค่าการ prediction ให้ได้ครบทุกรายวิชา โดยทางผู้จัดทำได้ทำการแก้ไขตามข้อสมมติฐานที่ได้ตั้งไว้ในครั้งก่อนว่าข้อผิดพลาดอาจเกิดขึ้นในส่วนสุดท้ายของ function ที่ทำการ loop + sorted ค่าของ prediction โดยทางผู้จัดทำได้ทำการเปลี่ยน algorithm ในการ loop นำค่าออกมา ผลลัพธ์ออกมาเป็นที่น่าพอใจโดยสามารถแก้ปัญหาได้ไปด้วยดี โดยรูปที่ 9 คือ algorithm ในการ loop แบบใหม่

```
for sub_id, grade in pred_ratings:
    if sub_id in subId_name:
        dic = {"subject_id" : sub_id, "sub_name" : subId_name[sub_id], "grade" : round(grade, 2)}
        response.append(dic)
return response
```

รูปที่ 9 function ที่ได้ทำการแก้ไข

ปัญหา ณ ปัจจุบัน

1. ปัญหาการ prediction ที่ไม่สมบูรณ์ เนื่องด้วยหลังจากการทดลองและตรวจสอบการ prediction ของ function ที่ทำการ optimize มาใหม่ พบว่าการทำนายนั้นเกิดข้อผิดพลาดที่ร้ายแรงขึ้น เช่น เมื่อมีการร้องขอการทำนาย 2 ครั้งโดยมีรายละเอียดดังนี้ ข้อมูลของผู้ใช้งานนั้นจะร้องขอการทำนายในรายวิชาเดียวกัน และ ผู้ร้องขอได้ให้ขอมูลเกรดของตนในรายวิชาเดียวกันแต่เกรดไม่เท่ากัน ตามหลักแล้วผลลัพธ์จำเป็นต้องออกมามีค่าที่แตกต่างกัน แต่ผลลัพธ์ที่ได้ปรากฏออกมาในรูปแบบเดียวกัน ซึ่งทางผู้จัดทำได้ตั้งข้อสมมติฐานว่าข้อผิดพลาดอาจเกิดจากการทำงาน 2 ส่วน ส่วนแรกคือการ process data ที่จะใช้ในการ train model ส่วนที่สองคือส่วนในการนำข้อมูลเข้า prediction

5. สิ่งที่จะดำเนินการต่อไป

1. ทำการพัฒนาโครงงานต่อตามแนวทางที่ได้ปรึกษากับที่ปรึกษา
2. จัดการปัญหาการ prediction ที่ไม่สมบูรณ์ โดยทางผู้จัดทำได้ทำการวางแนวทางในการแก้ไขไว้ว่าจะทำการตรวจสอบทุกการทำงานที่ละ process อย่างละเอียด
3. พัฒนา ในส่วนของ frontend