

System and Software Design Report

โครงการ Linked Out

เสนอ

ผศ.นคธิพย์ พร้อมพูล

รายชื่อผู้จัดทำ

ชาญชัย	รัตนะศิรากุล	6130115221
ธีรภัทร	ติระนาทวิทยาภูล	6130247421
ปุณยวัชร์	รุจิพิรานันท์	6131027921
พสวัฒน์	แตงใหญ่	6131029121
พสวัต	ไม้เหลือง	6131030721
ภูมิพัฒน์	จรจัด	6131037121
สรรัณย์	กฤตเวรนันท์	6131044521

โครงการนี้เป็นส่วนหนึ่งของรายวิชา 2110335 วิศวกรรมซอฟต์แวร์ 1

ภาคการศึกษาต้น ปีการศึกษา 2563

จุฬาลงกรณ์มหาวิทยาลัย

สารบัญ

1) Project name: Linked Out	5
2) Introduction and the to-be system overview.....	1
3) Objective of the project	1
4) Reference document	2
5) Objective of the design document	2
6) Design criteria and design activities.....	3
6.1 Design criteria: Coupling and Cohesion.....	3
6.2 Adding specifications.....	4
6.3 Restructuring the design: Factoring, Normalization	5
7) Introduction of an overview of system design modeling.....	10
8) System design constraint	11
9) Use case diagram.....	12
10) Class diagram.....	13
11) CRC class	14
12) Component diagram	16
13) CRUDE matrix.....	17
14) Contract specification (2 items).....	19
15) Method specification	20
16) Verifying and validating class and method design.....	21
17) User interface design principles.....	23
17.1 Layout.....	23
17.2 Content Awareness	23
17.3 Aesthetics.....	23
17.4. User Experience	24
17.5. Consistency.....	24
17.6. Minimal User Effort.....	25
18) Detail Real Use Case description.....	26
19) WND for Detail Real Use Case of topic 18	28
20) User interface design including relevant messages for WND of topic 18)	29
21) Nonfunctional requirements and their effect on physical architecture design.....	52

22) Overview of infrastructure design	53
Network Diagram	54
Deployment Diagram	55
Low-cost Deployment Modification	56
Low-cost Network Diagram.....	57
Low-cost Deployment Diagram	58
23) Hardware and software specification	59
24) Verifying and validating the physical architecture layer.....	60
25) Design Impact	60
26) Conclusion	61
System installation and operation.....	61
1.1) Conversion Plan	61
1.1.1) Hardware Purchase and Installation.....	61
1.1.2) Software Installation	61
1.1.3) Data Conversion	62
1.2) Conversion Dimension.....	62
1.2.1) Conversion Strategy	62
1.2.2) Conversion Style.....	62
1.2.3) Conversion Location	62
1.2.4) Conversion Modules	62
2.1.1) Online support.....	62
2.1.2) Admin support	63
2.2.1) Users Feedback.....	63
2.2.2) Closed for maintenance	63
2.3.1) Project team review.....	63
2.3.2) System review.....	63

สารบัญรูปภาพ

Figure 1 Class AdminPane.....	4
Figure 2 Class Recruiter(Old)	4
Figure 3 Class RecruiterPane (New).....	4
Figure 4 Package Member Encapsulation.....	5
Figure 5 User Factoring.....	6
Figure 6 After Normalizing	7
Figure 7 Before Normalizing	7
Figure 8 Normalized Banner.....	7
Figure 9 Package Diagram	8
Figure 10 Class Member.....	9
Figure 11 Class MemberController	9
Figure 12 Class MemberPane.....	10
Figure 13 Use Case Diagram	12
Figure 14 Class Diagram.....	13
Figure 15 Component Diagram.....	16
Figure 16 Verifying and Validating createChatroom	21
Figure 17 Verifying and Validating notifyRefusedMember	22
Figure 18 Layout	23
Figure 19 Content Awareness	23
Figure 20 Aesthetics	24
Figure 21 User Experience	24
Figure 21 WND Diagram	28
Figure 22 UI For Browse-Job Use Case.....	29
Figure 23 Home Page.....	30
Figure 24 Login Screen	31
Figure 25 Sign Up Screen	32
Figure 26 Sign Up Screen (2)	33
Figure 27 Sign up For Member Screen (After Clicking “Finding Job” in Figure 26)	34
Figure 28 Sign up For Recruiter Screen (After Clicking “Recruiter Labors” in Figure 26).	35
Figure 29 Success Sign Up Screen.....	36
Figure 30 Fail Sign Up Screen	37
Figure 31 Search Screen.....	38

Figure 32 Search Screen (After Clicking “Search” in Figure 31)	39
Figure 33 Requiring GPS access Screen (After Clicking “APPLY” in Figure 32)	40
Figure 34 Job Feeds	41
Figure 35 Applying job Screen (After Clicking “Apply Now” in Figure 34)	42
Figure 36 Applying Success Screen	43
Figure 37 Chat Notification	44
Figure 38 Chatroom Screen	45
Figure 39 Boosting Screen	46
Figure 40 Boosting Screen (2)	47
Figure 41 Payment Screen	48
Figure 42 Pay by internet banking	49
Figure 43 Pay by credit card	50
Figure 44 Buying Banner Screen	51
Figure 45 Network Diagram	54
Figure 46 Deployment Diagram	55
Figure 47 Low-cost Network Diagram	57
Figure 48 Low-cost Deployment Diagram	58

สารบัญตาราง

Table 1 RecruiterPane's CRC Card.....	14
Table 2 AdminPane's CRC Card	15
Table 3 CRUDE Matrix.....	18
Table 4 CreateChatroom Contract Specification.....	19
Table 5 NotifyRefuseMember Contract Specification	19
Table 6 NotifyRefusedMember Contract Specification.....	20
Table 7 CreateChatroom Contract Specification	20
Table 8 Browse-Job Use Case Desription.....	26
Table 9 Nonfunctional requirements and their effect on physical architecture design .	53
Table 10 Hardware and software specification	60
Table 11 List of teammate contribution	64

1) Project name: Linked Out

2) Introduction and the to-be system overview

เนื่องด้วยสถานการณ์การแพร่ระบาดของไวรัส COVID-19 ในช่วงเวลาที่ผ่านมาที่ก่อให้เกิดผลกระทบในหลายด้าน โดยเฉพาะด้านเศรษฐกิจ ซึ่งส่งผลให้เกิดการเลิกจ้างพนักงานทั้งพาร์ทไทม์ และพนักงานประจำเป็นจำนวนมาก โดยกลุ่มที่ได้รับผลกระทบโดยตรงคือกลุ่มพนักงานที่ถูกบริษัทเชิญออก และกลุ่มบัณฑิตใหม่ที่พึงจบการศึกษา

ทางคณะผู้จัดทำเล็งเห็นความสำคัญของการมีสื่อกลางในการช่วยจับคู่ระหว่าง ผู้ที่ต้องการจ้างงาน และผู้ที่ต้องการหางาน ที่มีศักยภาพพอจะช่วยให้กระบวนการหางานในช่วงการแพร่ระบาดของไวรัส COVID-19 เกิดขึ้นได้รวดเร็ว และลดโอกาสในการแพร่กระจายของเชื้อโรคจากการเดินทางเข้าไปติดต่อสถานที่ทำงานต่าง ๆ ได้มากขึ้น

คณะผู้จัดทำจึงจัดตั้งโครงการและพัฒนาเว็บแอปพลิเคชัน Linked Out ขึ้น โดยเว็บแอปพลิเคชันนี้จะทำหน้าที่เป็นสื่อกลางโดยมีระบบแนะนำงานที่ใกล้เคียงกับสถานที่ที่ผู้หางานเปิดใช้แอปพลิเคชันหรือตำแหน่งงานที่ผู้หางานสนใจทำงานในละแวกนั้น ระบบการสมัครสมาชิกและยืนยันตัวตนผู้ใช้งานระบบ เพื่อกรุ๊ปตุนให้ผู้ที่กำลังหางานได้เจอกับงานที่ตนเองต้องการ และกรุ๊ปตุนให้บริษัทที่กำลังรับสมัครพนักงานได้เจอกับผู้ที่มีคุณสมบัติตามต้องการได้รวดเร็วขึ้น

ระบบจะใช้ข้อมูลจากรูปแบบการทำงานจากเว็บไซต์ www.rgf-hragent.asia/th/thailand ซึ่งเป็นระบบที่เปิดให้ผู้สมัครค้นหางานได้ตามคำค้น คณะผู้จัดทำจึงนำแนวความคิดนี้มาปรับแก้ให้合ขึ้น โดยจะให้ความสำคัญกับระยะทางที่ใช้เดินทาง รวมทั้งพื้นที่ที่ทำงานด้วย โดยจะมีทางเลือกให้หางานตามอาชีพที่ผู้จัดหางานหรือสมาชิกทั่วไปต้องการสามารถค้นหางานในรัศมีที่กำหนดจากจุดที่กำหนดได้ และมีการหารายได้ผ่านการซื้อโฆษณาของผู้จัดหางานเพื่อเพิ่มอัตราการมองเห็นจากผู้สมัคร หรือการซื้อสื่อโฆษณาในตำแหน่งอื่น ๆ ในระบบ

3) Objective of the project

โครงการจัดทำนี้มีเป้าหมายเพื่อพัฒนาเว็บแอปพลิเคชันเพื่อเป็นตัวกลางสื่อกลางในการช่วยจับคู่ระหว่างผู้หางาน (ผู้ประสงค์จะสมัครงาน) และผู้จัดหางาน (ผู้ประสงค์จะรับสมัครพนักงาน) เพื่ออำนวยความสะดวกให้กระบวนการสมัครงานเป็นไปได้อย่างรวดเร็วและลดการเดินทางของกลุ่มคน โดยระบบที่จะพัฒนามีบริการดังต่อไปนี้

- จัดหางานที่ใกล้เคียงกับตำแหน่งของผู้ใช้งาน
- ยืนยันตัวตนผู้ใช้งานและผู้จัดหางาน
- ยืนยันการชำระเงินในระบบ
- ระบบยืนและส่งใบสมัครงาน
- การสนับสนุนภายในระบบ

4) Reference document

- System Requirements Specification Report โครงการ Linked Out (pg. 1-10)
- Project Proposal โครงการ Linked Out (pg. 1-25)
- Class Slides: Chapter [7] (pg. 1-46)
- Class Slides: Chapter [8] (pg. 1-56)
- Class Slides: Chapter [10] (pg. 1-56)
- Class Slides: Chapter [11] (pg. 1-56)

5) Objective of the design document

คณะกรรมการจัดทำได้ทำการอภิปรายบันทึกข้อความเพื่อเป็นส่วนหนึ่งของการพัฒนาระบบ Linked-Out โดยเอกสารฉบับนี้จะแสดงรายละเอียดของกระบวนการที่ใช้ในการออกแบบ วิธีการติดตั้งระบบ เพื่อให้มั่นใจว่าการออกแบบนั้นสอดคล้องกับความต้องการของระบบและผู้ใช้งาน และใช้เป็นเอกสารอ้างอิงในการพัฒนาระบบให้มีความเข้าใจตรงกัน โดยขั้นตอนการออกแบบมีดังนี้

- จัดทำเกณฑ์และหลักการในการออกแบบระบบ
- สำรวจความต้องการของผู้ใช้งานเพิ่มเติมและออกแบบแบบจำลองเชิงฟังก์ชันและคำอธิบายของระบบ

- 1) Use Case Diagram
- 2) Use Scenarios
- 3) Detail Real Use Case Description

- การออกแบบแบบจำลองเชิงโครงสร้าง แบบจำลองเชิงพฤติกรรมของระบบ Contract Specification และ Method Specification

- 1) Class Diagram
- 2) CRC Card
- 3) CRUDE Matrix
- 4) Contract Specification
- 5) Method Specification

- การตรวจสอบและตรวจสอบการออกแบบให้มีความสอดคล้องกัน
- จัดทำเกณฑ์และหลักการในการออกแบบส่วนต่อประสานผู้ใช้
- การออกแบบส่วนต่อประสานผู้ใช้ (User Interface)

- 1) Window Navigation Diagram
- 2) UI Design Prototype

- การออกแบบสถาปัตยกรรมระบบ ได้แก่'

- 1) Architecture Model
- 2) Green IT
- 3) Network Diagram

4) Deployment Diagram

5) Non-functional Requirement and Physical Architecture Design

- จัดทำข้อกำหนดมาตรฐานด้วย Entity Boundary และซอฟต์แวร์ของระบบ
- การตรวจสอบและตรวจสอบการออกแบบสถาปัตยกรรมของระบบ
- จัดทำผลกระทบของการออกแบบ
- จัดทำแผนการติดตั้งและการทำงานของระบบ

6) Design criteria and design activities

6.1 Design criteria: Coupling and Cohesion

ผู้พัฒนาได้เลือกใช้รูปแบบ Entity Boundary Control ในการออกแบบระบบ โดยมีการรวมให้การเรียกใช้ method ข้าม package จะต้องกระทำการผ่าน Control Class เท่านั้น โดยปรับแยก Application Logic ออกจาก Entity Class และแยกการปฏิสัมพันธ์กับผู้ใช้งานให้กระทำการผ่าน Boundary Class เท่านั้น ทั้งนี้เพื่อลด Coupling ระหว่าง Package และเพิ่ม Cohesion ระหว่าง package โดยมีรายละเอียดดังต่อไปนี้

6.1.1 ความคู่คบ (Coupling)

ผู้พัฒนาได้ลดความคู่คบ (Coupling) ของคลาสและเมธอดให้เหลือเท่าที่จำเป็นเท่านั้น โดยประเมินความคู่คบระหว่าง Package แสดงในตารางด้านล่าง

	Job	User	Admin	Banner	Chat	Log	Payment
Job	0						
User	2	0					
Admin	0	2	0				
Banner	0	0	2	0			
Chat	0	2	0	0	0		
Log	0	0	2	0	0	0	
Payment	0	0	2	0	0	0	0

หมายเหตุ: แต่ละหมายเลขที่แสดงในตารางข้างต้น มีความหมาย ดังนี้

0 = No Direct Coupling: เมธอดตั้งกล่าวไม่มีความเกี่ยวข้องกัน

1 = Data: มีการส่งค่าตัวแปร (Variable) ระหว่างเมธอด

2 = Stamp: มีการส่งค่าตัวแปรที่มีความซับซ้อน (Complex Variable) ระหว่างเมธอด ตัวอย่างเช่น

ระหว่าง Package Log และ Admin เป็นความคู่คบแบบ Stamp เนื่องจาก AdminPane มีการเรียกเมธอด getLog(,,LogFilter) ซึ่ง LogFilter ประกอบด้วยตัวแปรอย่างหลายชนิด จึงจัดเป็นตัวแปรที่มีความซับซ้อน

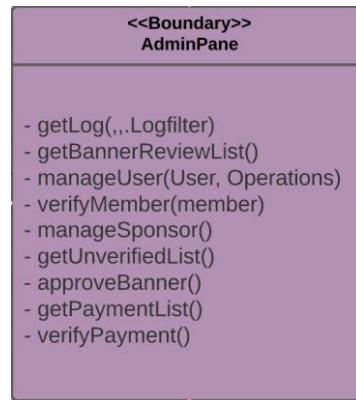


Figure 1 Class AdminPane

3 = Control: มีการส่งค่าตัวแปรที่ใช้ควบคุมการทำงาน (Control Variable) ระหว่างเมธอด

4 = Global: มีการเรียกใช้ตัวแปรส่วนกลาง (Global Variable) ระหว่างเมธอด

5 = Pathological: มีการเรียกใช้ค่าหรือเมธอดภายในของเมธอดที่ถูกเรียกใช้งาน

6.1.2) ความเชื่อมแน่น (Cohesion)

- Method Cohesion: พิจารณาแต่ละเมธอดให้ทำแค่ 1 อย่างเท่านั้น หากเมธอดใดทำหลายอย่างจะทำการแยกเป็นเมธอดย่อย ๆ
- Class Cohesion: พิจารณาแต่ละคลาส หากคลาสได้ Represent หลายอย่าง จะทำการแยกออกเป็นคลาสย่อย ๆ ที่ Represent แค่สิ่งเดียวเท่านั้น
- Generalization Cohesion: พิจารณาการสืบความหมายของคลาสต่าง ๆ ที่มีความสัมพันธ์กันแบบ Generalization (a-kind-of)

6.2 Adding specifications

ผู้พัฒนาได้ทำการทวนสอบ Class Diagram และ ปรับการออกแบบโดยอิงจาก ECN Plan ที่ได้กล่าวไว้ในหัวข้อ 6.1 โดยยังคงตอบสนองต่อทุกรูปแบบที่ระบุไว้ เช่น คลาส Recruiter ได้มีการออกแบบใหม่ ดังตัวอย่างต่อไปนี้

เดิม Recruiter มีหน้าที่ค่อนข้างคลุมเครือ ทั้งเก็บข้อมูลของผู้ว่าจ้างและมีเมธอดต่าง ๆ ในคลาสเดียวกัน จึงเปลี่ยนให้ Class Recruiter(เดิม) กลายเป็น Class RecruiterPane ซึ่งมีการเพิ่มเมธอด getJobApplicant() เพื่อรับกรณ์ใช้งานที่ผู้ว่าจ้างต้องการดูรายชื่อของผู้สมัครงาน รวมถึงกำหนดให้ Class ตั้งกล่าวเป็น Boundary ตามหลักของ ECN Plan และให้ตัว Entity เดิมปราศจาก logic ใด ๆ รวมถึงได้มีการทวนสอบการเรียกเมธอดให้ครบถ้วน ทั้งการ get การ set ต่าง ๆ

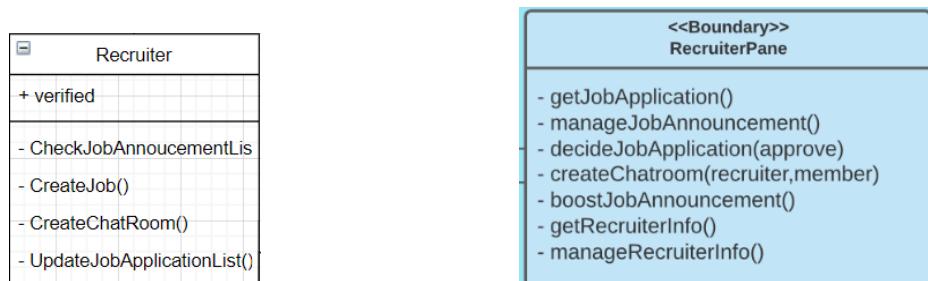


Figure 1 Class Recruiter(Old)

6.2.1 Encapsulation

สำหรับ method ที่ใช้ใน Control Class ที่ต้องให้ package อื่นเรียกใช้ ก็ทำเป็น public ไว้ แต่ถ้าเรียกใช้กันเองภายใน package ก็ให้เป็น default เพื่อความ Encapsulation ส่วน Entity ก็ให้เป็น private ทั้งหมดโดย assume ว่ามี getter และ setter ของแต่ละ attribute ไว้

สำหรับ Boundary Class ที่เป็น pane, assume ว่ามีการเรียกใช้เมธอดจาก component ภายใน เช่น เมื่อกดปุ่มก็ให้เรียกเมธอดซึ่งจะไปเรียกที่ Controller อีกต่อหนึ่ง จึงให้เป็น private (เพราะเรียกกันภายในเท่านั้น) ส่วน Boundary Class ที่เป็น Gateway ให้เป็น public เนื่องจากเป็น service ที่ต้องเรียกใช้จากภายนอก

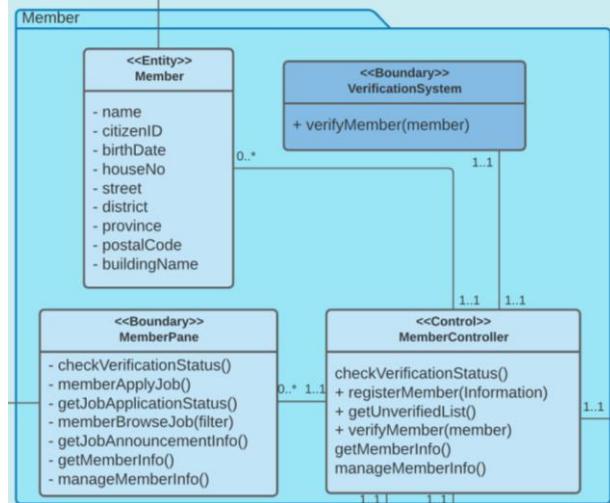


Figure 3 Package Member Encapsulation

6.2.2 Method Signature and Constraint

ได้มีการกำหนดชื่อเมธอดให้ใกล้เคียงกับชื่อ use case หรือ scenario มากรีด แล้วมีการกำหนด arguments, return type, และ constraints โดยสามารถดูได้ใน contract specifications

6.3 Restructuring the design: Factoring, Normalization

6.3.1 Factoring

มีการนำ Entity-Control-Boundary (ECB) เข้ามาปรับใช้ในการออกแบบ เมื่อมีการทวนสอบ Class Diagram ที่แก้ไขแล้วดังกล่าว พบร่วมกับข้อมูลบางส่วนที่มีความซ้ำซ้อน และสามารถจัดแยกให้ Member, Recruiter เป็น Generalization ของ User ได้เป็นไปตามหลัก ECN ดังตัวอย่างต่อไปนี้

เดิม Entity User มีการเก็บข้อมูลที่ครอบคลุมทั้งในกรณีที่ User เป็น Recruiter และกรณีที่ User เป็น Member ซึ่งเมื่อวิเคราะห์ดูแล้ว เราสามารถทำการ Generalize ออกมาระบบที่ Class Member และ Class Recruiter ได้

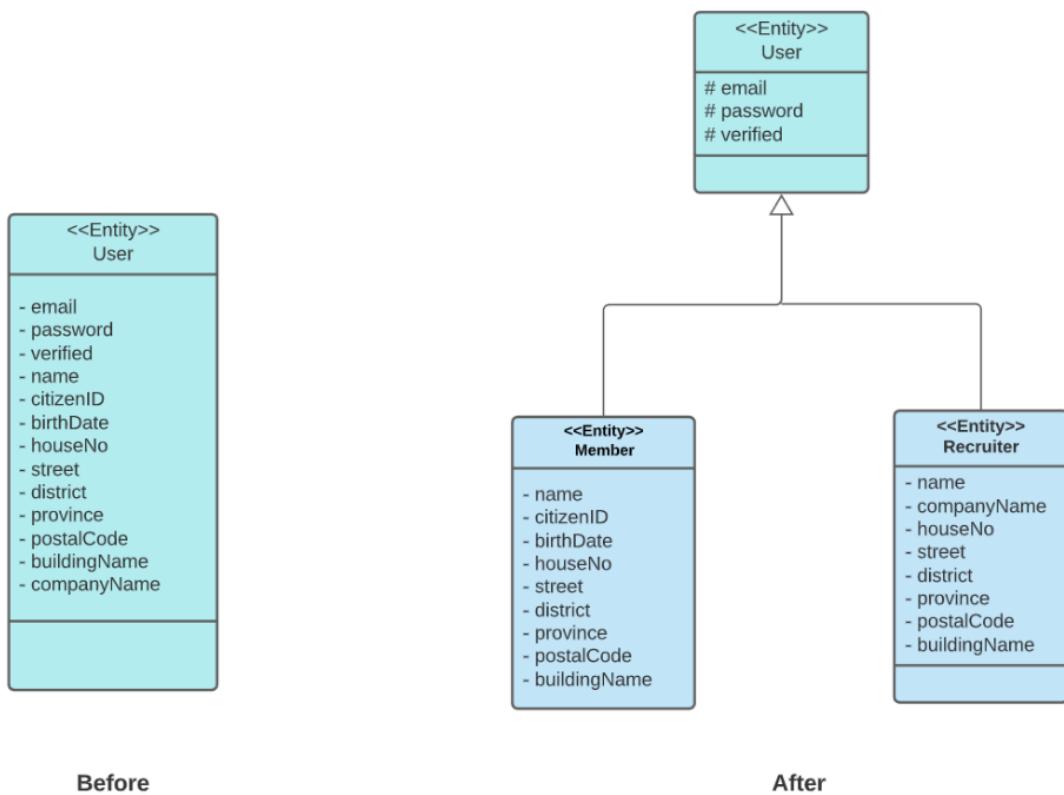


Figure 4 User Factoring

6.3.2 Normalization

ผู้พัฒนาได้ทำการ Normalize เพื่อเพิ่ม Cohesion ในระบบ โดยการแยก Class RequestBuyBanner ออกจาก Class Banner และกำหนดให้ RequestBuyBanner เป็นส่วนหนึ่งของ Banner รวมไปถึงลด Class ที่ไม่จำเป็นออกจาก Class Diagram เช่น ลด Class PaymentSlip เนื่องจากระบบบันทึกใช้การยืนยันการทำธุรกรรมผ่าน PaymentGateway ไม่จำเป็นต้องมี Class ดังกล่าว เป็นหลักฐานการทำธุรกรรม

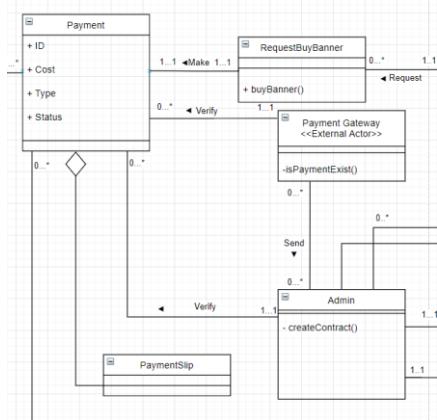


Figure 5 Before Normalizing

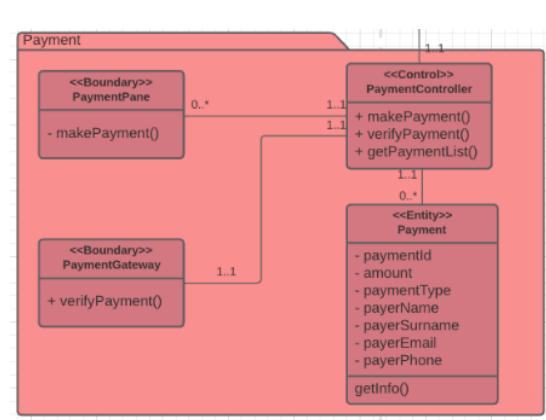


Figure 6 After Normalizing

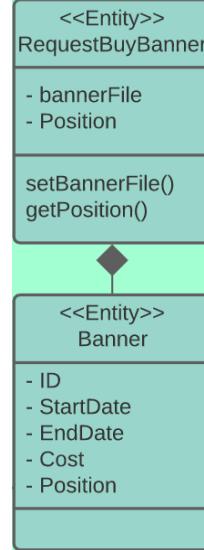


Figure 7 Normalized Banner

6.4 Package Diagram

ภายหลังจากที่ได้มีการออกแบบและจัดแบ่งคลาสใหม่ๆ ทุกคลาส โดยนำแต่ละคลาสมาจัดกลุ่มให้เป็นแพคเกจ (Package) เพื่อลดความซับซ้อนของแผนภาพที่ใช้ในการออกแบบระบบ โดยแบ่งตามความเกี่ยวข้องกัน (dependency) ซึ่งแบ่งได้ทั้งสิ้น 7 ประเภท และได้มีการกำหนด dependency ระหว่างแพคเกจเพื่อแสดงความสัมพันธ์ของแพคเกจเหล่านั้น ดังนี้ต่อไปนี้

1. Job package
2. User package
3. Admin package
4. Banner package
5. Chat package
6. Log package
7. Payment package

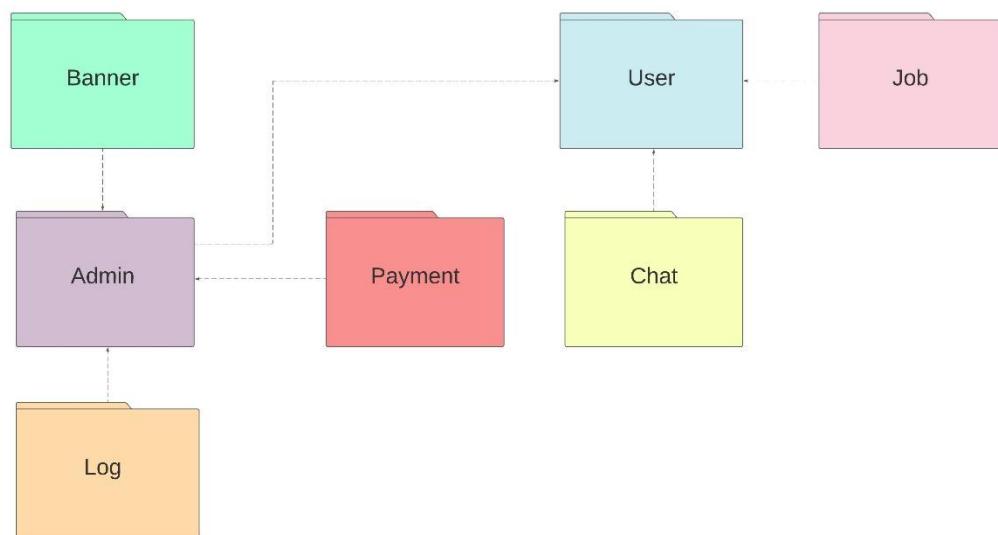


Figure 8 Package Diagram

Package Banner ขึ้นกับ Package Admin เนื่องจาก Admin เป็นผู้จัดการ Banner ในระบบทั้งหมด
 Package User ขึ้นกับ Package Admin เนื่องจาก Admin สามารถจัดการ User ในระบบทั้งหมด
 Package Payment ขึ้นกับ Package Admin เนื่องจาก Admin เป็นผู้ตรวจสอบ Payment ในระบบ
 Package Chat ขึ้นกับ Package User เนื่องจาก Recruiter และ Member เป็นผู้ใช้งานระบบ Chat
 Package Job ขึ้นกับ Package User เนื่องจาก Recruiter เป็นผู้ที่จัดการ Job ในระบบทั้งหมด และ Member เป็นผู้ที่จัดการการสมัคร Job ในระบบ
 Package Log ขึ้นกับ Package Admin เนื่องจาก Admin เป็นผู้ตรวจสอบ Log ของระบบ
 โดยรายละเอียดของ Class ที่อยู่ใน Package ต่าง ๆ สามารถดูเพิ่มเติมได้จากหัวข้อ

Figure12 Class Diagram ในหัวข้อ 10) Class Diagram

6.5 Layers

เพื่อให้ระบบที่ออกแบบสามารถใช้งานได้ตามความต้องการในโลกจริงซึ่งมีการติดต่อ กับระบบอื่น ๆ เป็นจำนวนมาก ผู้พัฒนาจึงได้ทำการวิเคราะห์โดยใช้ Layers ซึ่งสามารถแบ่งได้เป็น 5 layers ได้แก่

- 1) Foundation Layer คือ คลาสที่ทำหน้าที่จัดเก็บข้อมูลพื้นฐาน เช่น Primitive Data Types, Containers ซึ่งภาษาที่ใช้พัฒนารองรับอยู่แล้ว ยกตัวอย่างเช่น ใน Entity Member ประกอบด้วย
 - name (ชื่อ) จัดเก็บข้อมูลแบบแຄอักชระ (String)
 - citizenID (เลขบัตรประจำตัวประชาชน) จัดเก็บข้อมูลด้วยตัวเลข (Integer)
 - birthdate (วันเกิด) จัดเก็บด้วยข้อมูลแบบวันที่ (Date)
 - houseNo (เลขที่บ้าน) จัดเก็บด้วยข้อมูลแบบตัวเลข (Integer)
 - street (ถนน) จัดเก็บข้อมูลแบบแຄอักชระ (String)
 - province (จังหวัด) จัดเก็บข้อมูลแบบแຄอักชระ (String)
 - postalCode (รหัสไปรษณีย์) จัดเก็บข้อมูลแบบตัวเลข (Integer)
 - buildingName (ชื่ออาคาร) จัดเก็บข้อมูลแบบแຄอักชระ (String)



Figure 10 Class Member

- 2) Data Access and Management Layer: คลาสที่ทำหน้าที่เกี่ยวกับการจัดเก็บและแก้ไขข้อมูล เรียกว่า Data Access and Manipulation (DAM) class ซึ่งสามารถดูรายละเอียดการออกแบบ ได้จากหัวข้อ 23) Hardware and software specification
- 3) Problem Domain Layer: คลาสที่ระบบต้องใช้เพื่อตอบสนองความต้องการของระบบ ซึ่งทางผู้พัฒนาได้ทำการออกแบบให้ครอบคลุมแล้วจากเอกสารออกแบบนี้ เช่น MemberController

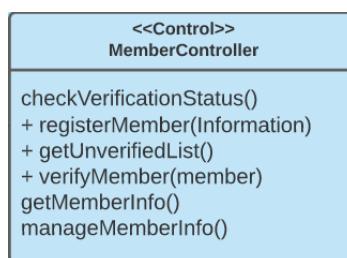


Figure 11 Class MemberController

- 4) Human-Computer Interaction Layer: คลาสที่ทำหน้าที่เกี่ยวกับการประสานงานระหว่างผู้ใช้งานกับระบบ
- 5) Physical Architecture Layer: คลาสที่ทำหน้าที่เกี่ยวกับการติดต่อสื่อสารระหว่าง ซอฟต์แวร์ระบบปฏิบัติการ และเครือข่าย เช่น MemberPane เป็น Class ที่อยู่ติดต่อกับ Frontend ที่รับคำสั่งหรือ Input ต่าง ๆ มาจากคนจริง ๆ อีกทีหนึ่ง

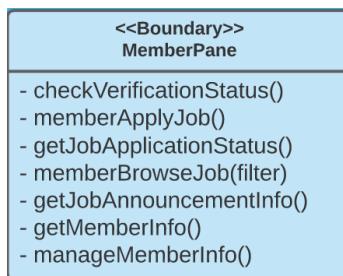


Figure 12 Class MemberPane

7) Introduction of an overview of system design modeling

ในเอกสารนี้ ในส่วนของข้อกำหนดการออกแบบระบบและซอฟต์แวร์เราได้ใช้วิธีการแบบ OOSAD เพื่อพัฒนา To-be system เพราะว่าเราต้องการสร้าง To-be system ที่มีความเชื่อมแน่นสูง ความคุ้คุบตា และสามารถแบ่งแยกเป็นวัตถุ (Object) ต่าง ๆ ได้ ซึ่งวิธีการนี้ยังสามารถแสดงให้เห็นถึงความสัมพันธ์ระหว่างวัตถุ ทำการสื่อสารกับทีมอื่นง่ายขึ้น โดยเฉพาะหากใช้มาตรฐาน UML

คณะกรรมการเลือกใช้ UML (Unified Modeling Language) ในการแสดงถึงระดับของนามธรรม (Level of abstraction) ของการพัฒนาระบบและซอฟต์แวร์ในรูปแบบของโมเดล เช่น Use-case diagram, Deployment diagram หรือ Component diagram เป็นต้น นอกจากนี้ UML ยังช่วยทำให้เห็นภาพและบันทึกโมเดลของระบบได้ง่ายขึ้น รวมไปถึงการออกแบบและโครงสร้างที่ทำให้ได้ครบตามข้อกำหนด ดังนั้นเรารึ่งได้ทำการเปรียบเทียบข้อดีและข้อเสียจากการใช้ UML ในการวิเคราะห์และการออกแบบ ดังแสดงในตารางต่อไปนี้

ข้อดี	ข้อเสีย
ทำให้สามารถจัดการซอฟต์แวร์ที่มีความซับซ้อนได้อย่างมีประสิทธิภาพจากการมองเป็นหน่วยย่อย ๆ	ไม่สามารถแสดงการเชื่อมต่อระหว่างวัตถุทั้งหมดในแผนภาพเดียว
หลักการห่อหุ้มและการซ่อนข้อมูลช่วยให้นักพัฒนาสร้างระบบที่ไม่ถูกก่อความโดยส่วนอื่น ๆ ของระบบ	ฟังก์ชันการทำงานถูกจำกัดอยู่ภายในวัตถุเท่านั้น ซึ่งอาจจะเป็นเรื่องยากในระบบที่มีขั้นตอนซับซ้อนหรือการคำนวณซับซ้อน
ทำให้การขยายระบบนั้นง่ายกว่าในระบบที่ใช้หลักการวิเคราะห์โครงสร้าง	ไม่สามารถรู้ได้ว่าต้องใช้วัตถุใดจึงจะสามารถสร้างการออกแบบระบบที่เหมาะสมสมที่สุด
มุ่งเน้นไปที่ข้อมูลมากกว่าขั้นตอน	โมเดลเชิงวัตถุไม่สามารถสนับสนุนถึงการติดต่อสื่อสารระหว่างวัตถุได้อย่างชัดเจน

8) System design constraint

8.1) ด้านแพลตฟอร์ม

8.1.1) ระบบเป็นเว็บแอปพลิเคชัน ทำให้จำเป็นต้องเขื่อมต่ออินเทอร์เน็ตผ่านอุปกรณ์คอมพิวเตอร์ หรือโทรศัพท์มือถือมีเว็บบราว์เซอร์ เช่น Microsoft Edge, Mozilla Firefox, Apple Safari, Opera Browser และ Google Chrome เป็นต้น

8.1.2) ระบบต้องใช้โพรโทคอล HTTPS ในการรับ-ส่งข้อมูลได้ ๆ จากบุคคลใด ๆ ที่เข้ามาใช้งานระบบ

8.1.3) ระบบจำเป็นต้องใช้ API ของระบบภายนอก ในระบบการจ่ายเงิน ระบบระบุตำแหน่งสมาชิก ระบบตรวจสอบสมาชิกและผู้จัดหางาน

8.1.4) การพัฒนาระบบเป็นการใช้พัฒนาแบบ cloud - based

8.2) ความสามารถและข้อจำกัดของผู้ใช้งาน

8.2.1) ระบบต้องให้ผู้ใช้งานระบบยืนยันตัวตนก่อนการใช้งานทุกรอบ

8.2.2) ผู้ใช้งานระบบจำเป็นต้องลงทะเบียนด้วย email และ password ก่อนใช้งานระบบ

8.2.3) ระบบใช้ผู้ดูแลระบบในการยืนยันการจ่ายเงินของผู้ใช้งานระบบที่ได้รับข้อมูลจาก payment gateway

8.3) ความปลอดภัย

8.3.1) ระบบจะร้องขอเข้าถึงข้อมูลของผู้ใช้เท่าที่จำเป็นและเหมาะสมต่อระบบเท่านั้น

8.3.2) ระบบต้องเก็บข้อมูลรหัสผ่านของผู้ใช้งานระบบในรูปแบบแฮช

8.3.3) ระบบต้องปกป้องบุคคลใด ๆ ที่เข้ามาใช้งานระบบจาก CSRF และ XSS

8.3.4) การใช้งานผ่านระบบโทรศัพท์มือถือไม่ควรถูก decompile ได้โดยง่าย

8.4) ด้านเวลาที่ใช้ในการทำงาน

8.4.1) การวิเคราะห์และออกแบบระบบจะต้องเสร็จภายในเดือน พฤศจิกายน

8.5) พื้นที่ให้บริการ

8.5.1) ระบบจะให้บริการเฉพาะสมาชิกและผู้จัดหางานที่มีสัญชาติไทยเท่านั้น

8.5.2) ระบบจะให้บริการรายในประเทศไทยเท่านั้น

8.6) ด้านกฎหมาย

8.6.1) ระบบต้องไม่ละเมิดข้อกฎหมายใดในพระราชบัญญัติว่าด้วยการการกระทำผิดเกี่ยวกับ

คอมพิวเตอร์ พ.ศ. 2560

9) Use case diagram

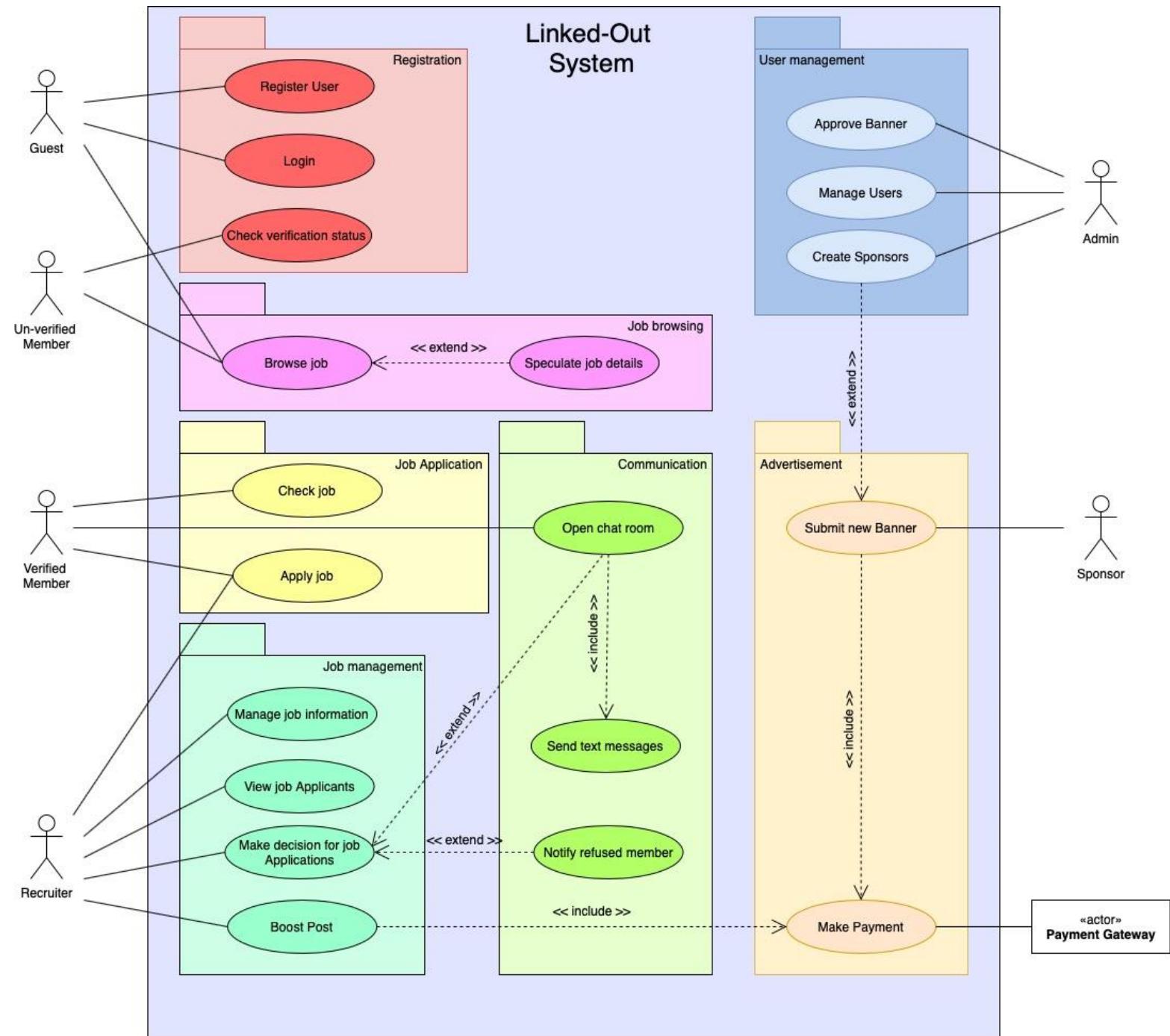


Figure 13 Use Case Diagram

10) Class diagram

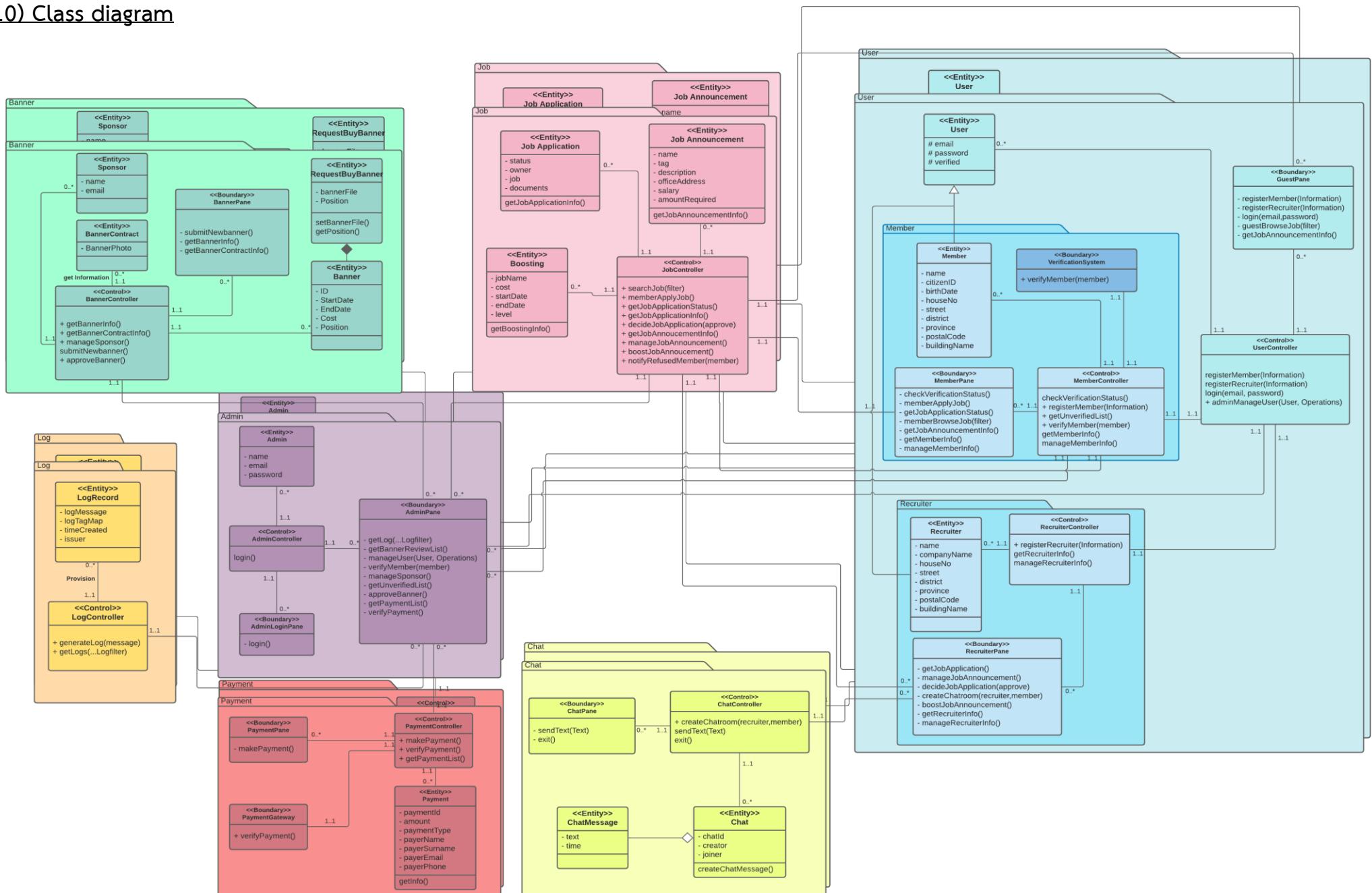


Figure 14 Class Diagram

11) CRC class

Front of CRC Card

Class Name: RecruiterPane	ID: I04	Type: Concrete Domain			
Description: A Pane of an individual or a represent of company who want to hire people to do work	Associated Uses Cases: Manage job announcement, View job Applicant, Make decision for job Application				
<table> <thead> <tr> <th>Responsibilities</th> <th>Collaborator</th> </tr> </thead> <tbody> <tr> <td> <ul style="list-style-type: none"> • View job application • Manage job announcement • Decide Job application • Create chat room • Boost job announcement • View recruiter information • Manage job announcement </td> <td> <ul style="list-style-type: none"> • JobController • JobController • JobController • ChatController • JobController • RecruiterController • JobController </td> </tr> </tbody> </table>		Responsibilities	Collaborator	<ul style="list-style-type: none"> • View job application • Manage job announcement • Decide Job application • Create chat room • Boost job announcement • View recruiter information • Manage job announcement 	<ul style="list-style-type: none"> • JobController • JobController • JobController • ChatController • JobController • RecruiterController • JobController
Responsibilities	Collaborator				
<ul style="list-style-type: none"> • View job application • Manage job announcement • Decide Job application • Create chat room • Boost job announcement • View recruiter information • Manage job announcement 	<ul style="list-style-type: none"> • JobController • JobController • JobController • ChatController • JobController • RecruiterController • JobController 				

Back of CRC Card

Attributes:
Relationships: Generalization (a-kind-of): Aggregation (has-parts): Other Association: JobController, Chatcontroller, Recruitercontroller

Table 1 RecruiterPane's CRC Card

Front of CRC Card

Class Name: AdminPane	ID: I03	Type: Concrete Domain
Description: A pane that people who manages or organizes the software.	Associated Use Cases: Approve Banner, Manage Users, Manage Sponsor	
Responsibilities <ul style="list-style-type: none"> ● Access Log record ● Access Banner review list ● Access user account ● Verify unverified member account ● Access unverified member list ● Approve banner ● Verify payment ● Manage sponsor ● Access Payment list 		Collaborator <ul style="list-style-type: none"> ● LogController ● AdminController ● UserContoller ● MemberController ● MemberController ● AdminController ● PaymentController ● AdminController ● PaymentController

Back of CRC Card

Attributes:
Relationships: <p>Generalization (a-kind-of):</p> <p>Aggregation (has-parts):</p> <p>Other Association: AdminController, MemberController, UserController, PaymentController, LogController</p>

Table 2 AdminPane's CRC Card

12) Component diagram

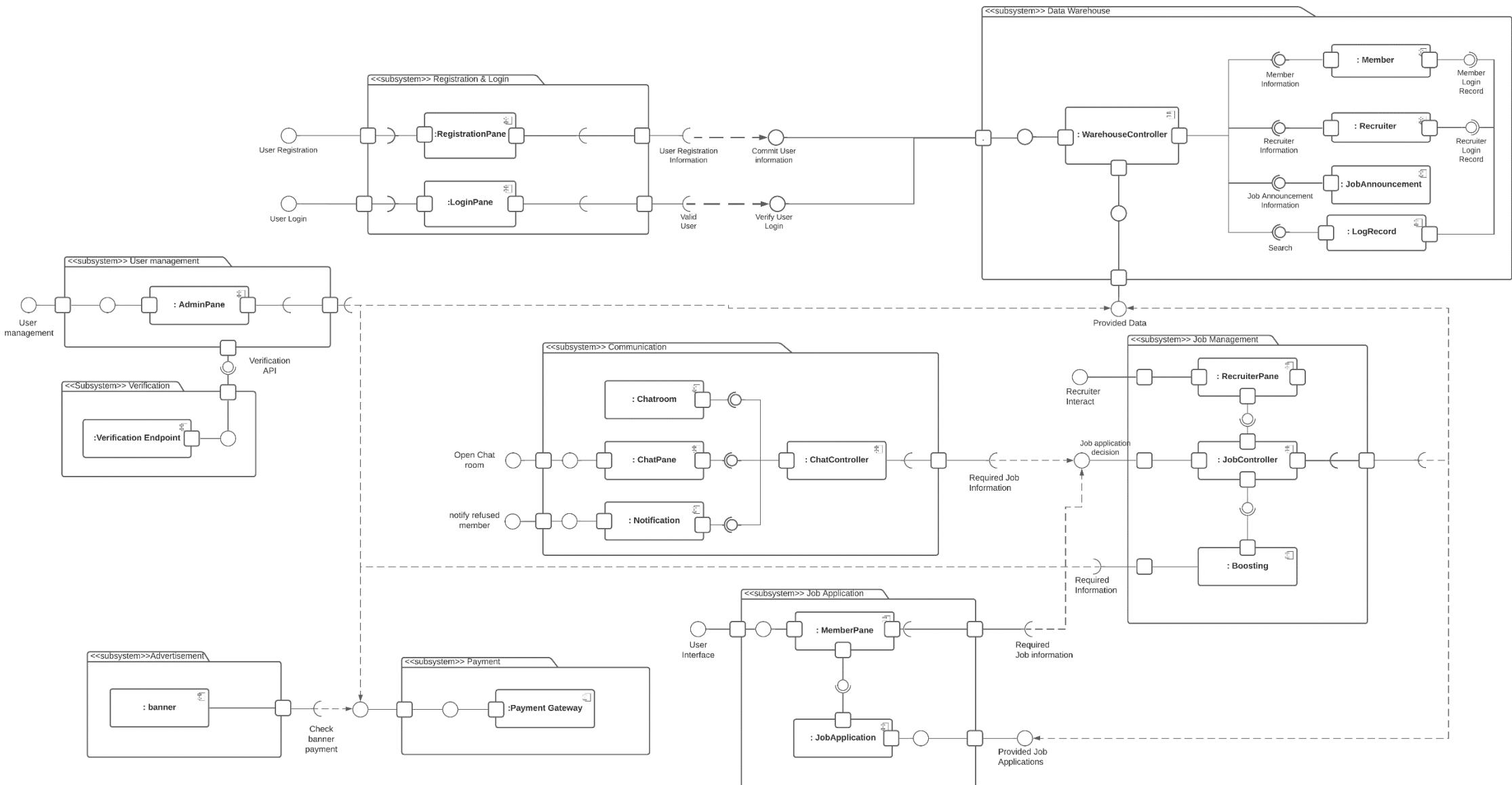


Figure 15 Component Diagram

13) CRUDE matrix

เราได้ทำการเลือก Use Case BrowseJob มาแสดง CRUDE Matrix

		MemberController	RecruiterPane	RecruiterController	JobController	ChatPane	ChatController	PaymentPane	PaymentController	AdminPane	AdminController	BannerPane	BannerController	BannerContract	RequestBuyBanner	LogController	LogRecord
Sponsor																	
Banner																	
Admin																	
Payment Gateway																	
Job Application																	
Job Annoucement				R,E													
Boosting																	
Chatroom																	
ChatMessage																	
Verified Member																	
Unverified Member																	
Guest																	
Recruiter																	
User																	
GuestPane				E													
UserController																	
VerificationSystem																	
MemberPane				E													
MemberController																	
RecruiterPane																	
RecruiterController																	
JobController																	
ChatPane																	
ChatController																	
PaymentPane																	
PaymentController																	
AdminPane																	
AdminController																	
BannerPane																	
BannerController																	
BannerContract																	
RequestBuyBanner																	
LogController																	
LogRecord																	

Table 3 CRUDE Matrix

14) Contract specification (2 items)

Method Name:	CreateChatroom	Class Name:	RecruiterPane	ID:	1
Clients :	ChatRoom, VerifiedMember, Recruiter				
Associated Use Cases:					
	Open Chat Room				
	Send text message				
	Make decision for job Applications				
Description of Responsibilities:					
	Called when a Recruiter wants to create a chat to schedule a meeting with a VerifiedMember.				
	This method creates a chat room with 2 participants.				
Arguments Received:					
	Member				
	Recruiter				
Type of Value Returned:					
	aChatRoom:ChatRoom				
Pre-Conditions:					
	Member is a member of Class Member				
Post-Cor					
	Chatroom is a member of Class Chatroom				

Table 4 CreateChatroom Contract Specification

Method Name:	NotifyRefuseMember	Class Name:	JobController	ID:	2
Clients :	MemberPane, MemberController				
Associated Use Cases:					
	Notify refused member				
	Make decision for job Applications				
Description of Responsibilities:					
	Called when a Recruiter rejects a Member's JobApplication.				
	This method will notify Member that their JobApplication is refused				
Arguments Received:					
	Member				
Type of Value Returned:					
	Null				
Pre-Conditions:					
	JobApplication.Owner is Member				
	Member is a member of Class Member				
	JobApplication.status is rejected (False)				
	JobApplication is a member of Class JobApplication				
Post-Cor					
	JobApplication is no longer a member of Class JobApplication				

Table 5 NotifyRefuseMember Contract Specification

15) Method specification

Method Name	CreateChatroom	Class Name :	RecruiterPane	ID:	1
Contract ID:	1	Programmer:	Sorathan	Date Due:	
Programming Language	Java				
Trigger/Events:					
	Recruiter wants to contact a job applicant(Member).				
Arguments Received:					
Data Type	Note				
Member	Correspondant of the chat.				
Recruiter	Correspondant of the chat.				
Messages Sent & Arguments Passed:	Data Type	Notes			
Recruiter.createChatRoom(recruiter,member)	chatRoom				
Arguments Returned:					
Data Type	Notes				
ChatRoom	Return the newly created chatroom				
Algorithm Specification:					
	chatRoom.createRoom()				
	chatRoom.setInitiator(Recruiter)				
	chatRoom.setCreationDate(time)				
	chatRoom.addParticipants(Member)				
Misc. Notes:					
	None.				

Table 7 CreateChatroom Contract Specification

Method Name	NotifyRefusedMember	Class Name :	JobController	ID:	2
Contract ID:	2	Programmer:	Sorathan	Date Due:	
Programming Language	Java				
Trigger/Events:					
	Recruiter reject a job application.				
Arguments Received:					
Data Type	Note				
Member	The owner of the job application.				
Messages Sent & Arguments Passed:	Data Type	Notes			
RecruiterPane.decideJobApplication(approve)	Boolean				
JobController.notifyRefusedMember()	String				
MemberController.notifyRefusedMember()	String				
MemberPane.notifyRefusedMember()	String				
Arguments Returned:					
Data Type	Notes				
Null					
Algorithm Specification:					
	RecruiterPane.decideJobApplication(approve)				
If(Approve == rejected) {					
	JobController.notifyRefusedMember()				
	MemberController.notifyRefusedMember()				
	MemberPane.notifyRefusedMember() }				
Misc. Notes:					
	None.				

Table 6 NotifyRefusedMember Contract Specification

16) Verifying and validating class and method design

Class Diagram and Method Specification

ผู้พัฒนาได้ทวนสอบและตรวจสอบให้ข้อมูลของระบบมีความตรงกันมากที่สุด ในตัวอย่างต่อไปนี้ มีการตรวจสอบระหว่าง Method Specification และ Class Diagram ว่ามีชื่อเมธ็อด, ชื่อ Class และ Data type ของ parameter ที่ตรงกัน

- Method CreateChatroom and Class RecruiterPane

		Method Name: CreateChatroom	Class Name : RecruiterPane	ID: 1		
Contract ID:		1	Programmer: Soralhan	Date Due:		
Programming Language Java						
Trigger/Events:		Recruiter wants to contact either a job applicant.				
<>Boundary<> RecruiterPane		Arguments Received:				
<ul style="list-style-type: none"> - getJobApplication() - manageJobAnnouncement() - decideJobApplication(approve) - createChatroom(recruiter,member) - postJobAdvertisement() - getRecruiterInfo() - manageRecruiterInfo() 		Data Type	Note			
		Member	Correspondant of the chat.			
		Recruiter	Correspondant of the chat.			
Messages Sent & Arguments Passed:		Data Type	Notes			
Recruiter.createChatRoom(recruiter,member)		chatRoom				
Arguments Returned:		Data Type	Notes			
		ChatRoom	Return the newly created chatroom			
Algorithm Specification:						
		chatRoom.createRoom()				
		chatRoom.setInitiator(Recruiter)				
		chatRoom.setCreationDate(time)				
		chatRoom.addParticipants(Member)				
Misc. Notes:						
		None.				

Figure 16 Verifying and Validating createChatroom

- Method NotifyRefusedMember and Class JobController

ภายหลังการทวนสอบ Method NotifyRefusedMember ซึ่งเป็นเมธอดที่เรียกใช้เพื่อแจ้งเตือนผู้สมัครงาน หากผู้รับสมัครไม่รับสมัครงาน ซึ่งรับ parameter 1 ตัวคือ Member ที่ต้องการแจ้งเตือน สอดคล้องกันทั้งใน Method Specification และ Class Diagram

Method Name:	NotifyRefusedMember	Class Name:	JobController	ID:	2						
Contract ID:	2	Programmer:	Sorathan	Date Due:							
Programming Language: Java											
Trigger/Events:											
Recruiter reject a job application.											
Arguments Received:											
Data Type:		Note:									
Member		The owner of the job application.									
Messages Sent & Arguments Passed:											
RecruiterPane.decideJobApplication(approve)		Data Type:		Notes:							
Boolean											
JobController.notifyRefusedMember()											
String											
MemberController.notifyRefusedMember()											
String											
MemberPane.notifyRefusedMember()											
String											
Arguments Returned:											
Data Type:		Notes:									
Null											
Algorithm Specification:											
RecruiterPane.decideJobApplication(approve)											
If(Approve == rejected)	JobController.notifyRefusedMember()										
Misc. Notes:											
None.											

Figure 17 Verifying and Validating notifyRefusedMember

17) User interface design principles

17.1 Layout

ผู้พัฒนาได้แบ่งส่วนของหน้าจอออกเป็น 2 ส่วน ได้แก่ Header Area และ Body Area แสดงถึงหัวข้อปัจจุบันที่ ทำงานอยู่ และแสดงรายละเอียดของหัวข้อนั้น

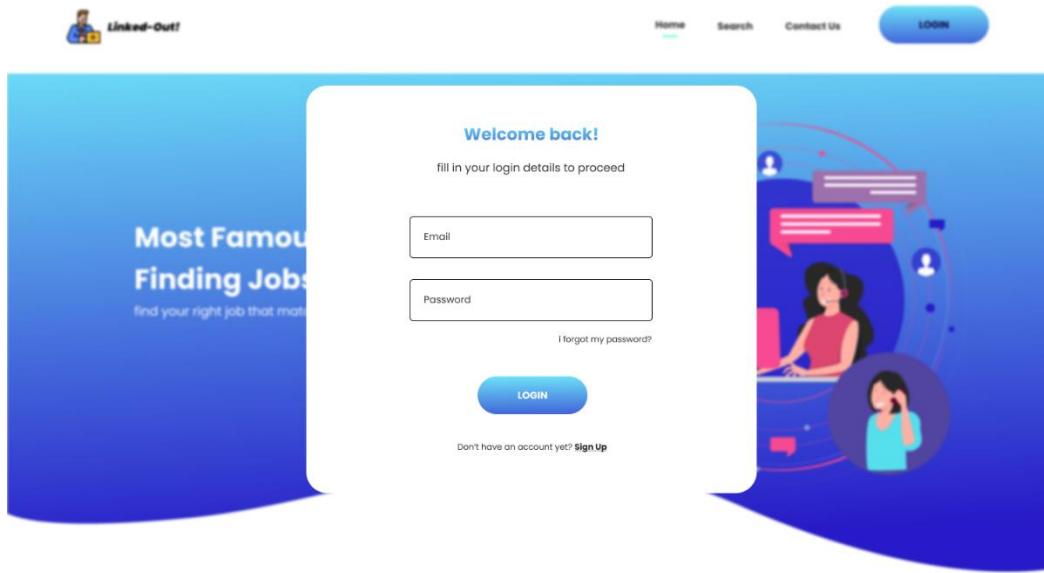


Figure 18 Layout

17.2 Content Awareness

ภายในเว็บไซต์ มีการใช้เนื้อหาที่ถูกต้อง ครบถ้วน สมบูรณ์ เพื่อให้ผู้ใช้เกิดความเข้าใจในเว็บไซต์มากยิ่งขึ้น เช่น label ในปุ่มบนแถบ menu บริเวณ header (Home, Search, Contact us)

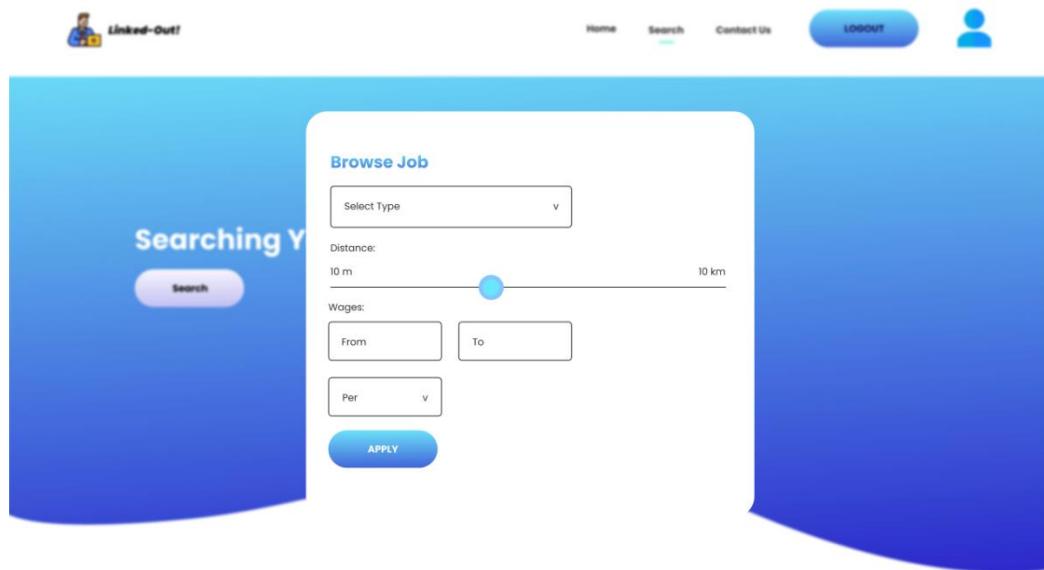


Figure 19 Content Awareness

17.3 Aesthetics

เว็บไซต์ใช้สีพื้นหลังสีฟ้าตัดกับสีตัวอักษรขาวเพื่อให้ง่ายต่อการอ่าน ใช้การออกแบบอย่างเรียบง่าย ไม่รกหน้าจอ

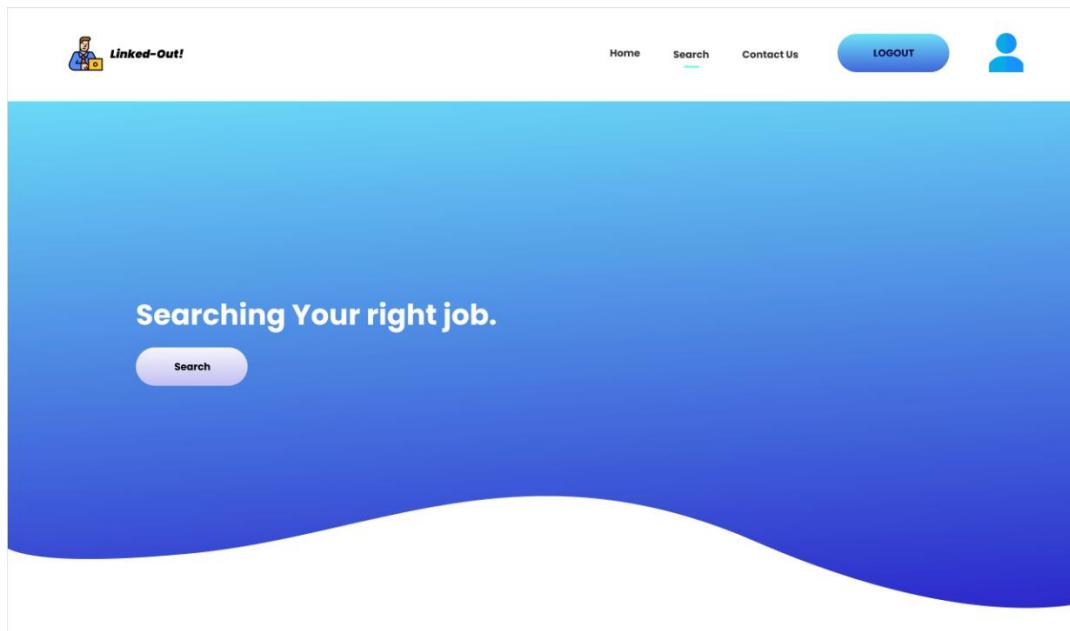


Figure 20 Aesthetics

17.4. User Experience

การออกแบบที่วาง layout ของหน้าต่าง ๆ ไว้ชัดเจน เน้นความพยายามในการใช้งาน มีการบอกถึง หัวข้อการทำางานนั้น ๆ และมีการแจ้งเตือนเมื่อเกิดปัญหาการใส่ข้อมูลที่ผิดหรือการแจ้งรายละเอียด เพิ่มเติมในหัวข้อนั้น ๆ ทำให้ง่ายต่อการใช้งานและสามารถเรียนรู้การใช้งานได้อย่างรวดเร็ว

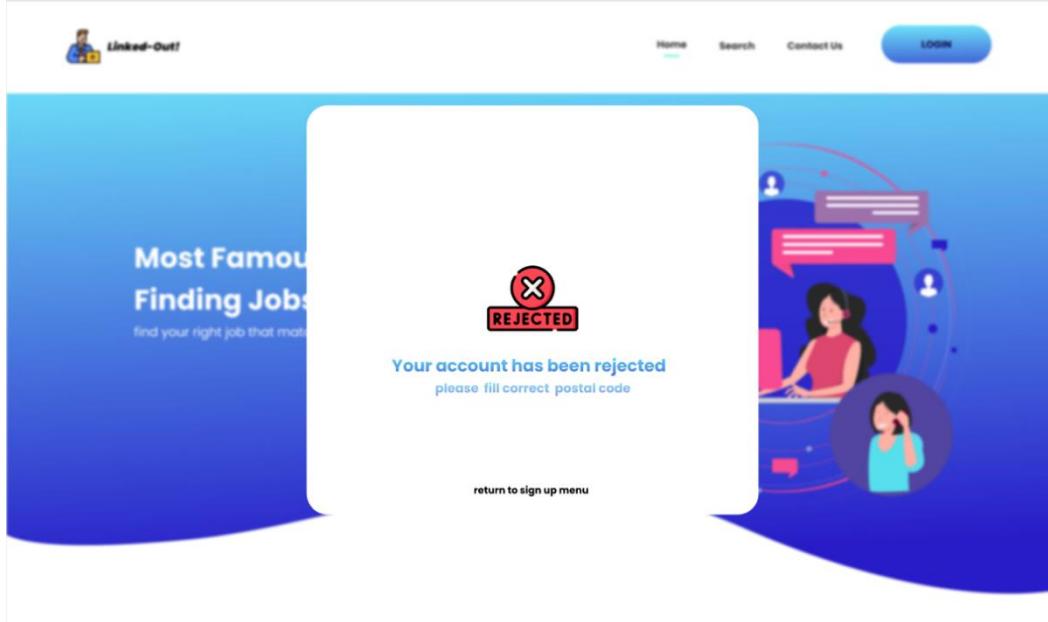


Figure 21 User Experience

17.5. Consistency

ทุก ๆ หน้าจะใช้ Layout แบบเดียวกัน โดยมีส่วน Header Area แสดงหัวข้อที่กำลังใช้งานอยู่ และทุก ๆ หน้าจะใช้สีในธีมเดียวกัน ทุก ๆ หน้าจะเป็นไปในทิศทางเดียวกัน ทำให้ผู้ใช้มีความเข้าใจถึง pattern ของเว็บไซต์มากยิ่งขึ้น

17.6. Minimal User Effort

ในส่วนของหน้าจอจะแสดงส่วนต่าง ๆ ของเว็บไซต์และสามารถเข้าถึงได้ทันที โดยจะขึ้น เส้นใต้หัวข้อของส่วนที่ผู้ใช้งานเข้าถึงอยู่ในปัจจุบัน

18) Detail, Real Use-case Description

Use Case Name: Browse Job	ID: I01	Importance Level: High		
Primary Actor: Guest, Un-verified Member, Verified Member	Use Case Type: Detail, Real			
Stakeholders and Interests:				
<p>Guest – wants to search for jobs of their interests using various filter.</p> <p>Un-verified – wants to search for jobs of their interests using various filter.</p> <p>Member – wants to search for jobs of their interests using various filter.</p>				
Brief Description: This use case describes how to browse a job in our application. User can apply the filter to find jobs that are more relevant to them.				
Trigger: member start searching for a nearby job.				
Type: External				
Relationships:				
Association: Guest, Un-verified Member, Verified Member.				
Include: -				
Extend: Speculate Job Details				
Generalization: -				
Precondition: Un-verified and member need to authenticate themselves beforehand.				
Postcondition: -				
Normal Flow of Events:				
<ol style="list-style-type: none"> 1. Member, guest, Un-verified clicks on the Searching button. 2. The System provides the Browse Job Form which contains job types drop down list, distance slider, and wage textbox. 3. Member, guest, Un-verified fill a browse job form which contains job types, distance and wage range. If GPS Access is in the form, Execute S-1: 4. The System provides Related Job Window If user clicks apply button from the list shown. Go to use case <i>Speculate Job Details</i>. 				
SubFlows:				
S-1: Request for GPS Access:				
<ol style="list-style-type: none"> 1. Prompt for GPS Access from user 				
Alternate/Exceptional Flows:				
S-1, 1a1: User clicks the Don't Allow button to deny their GPS Location. The System provides the Browse Job window (Use scenario : Member cancels GPS access).				

Table 8 Browse-Job Use Case Description

Use scenario: Member cancels GPS access	Use scenario: Member allows GPS access
1. Member browse an interested job by applying filter (1)	1. Member browse an interested job by applying filter (1)
2. System request access GPS location (S-1)	2. System request access GPS location (S-1)
3. Member does not permit the system to access their GPS location.(S-1,1a1)	3. Member grant access to GPS location (S-1)
4. Member go back to select new filter(1)	4. Member see through the queried jobs (2)

19) WND for Detail Real Use Case of topic 18

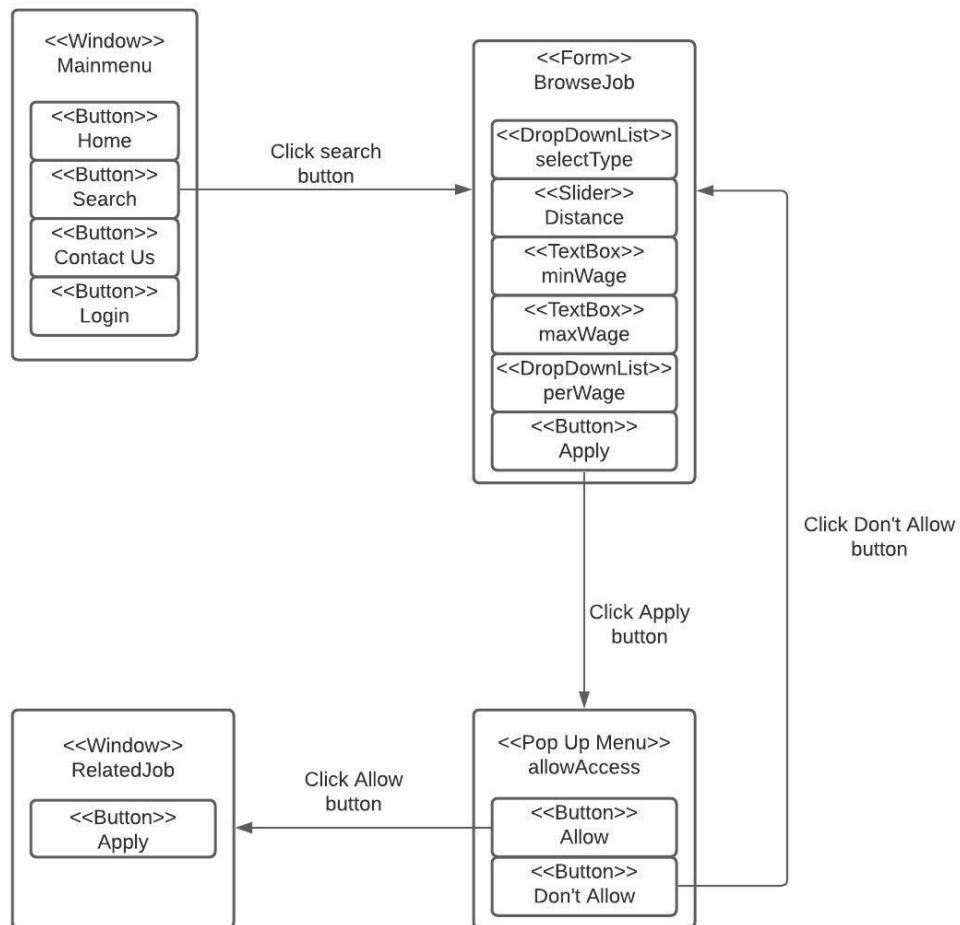


Figure 22 WND Diagram

Figure 23 UI For Browse-Job Use CaseFigure 21 WND Diagram

หลังจากที่เรากดปุ่ม Search ที่หน้าต่าง Main Menu ระบบก็จะนำไปสู่ที่แบบฟอร์ม Browse Job จากนั้นให้ user กรอกรายละเอียดตามแบบฟอร์ม เมื่อกดปุ่ม Apply จะมีข้อความแจ้งเตือนขออนุญาตเข้าถึงตำแหน่งที่ต้องของ user หาก user กดปุ่ม Don't Allow จะกลับไปที่แบบฟอร์ม Browse Job ให้กรอกรายละเอียดใหม่ แต่หากกดปุ่ม Allow จะไปที่หน้าต่าง Related Job ซึ่งจะปรากฏงานทั้งหมดที่ตรงตามที่ user กรอกรายละเอียดในฟอร์ม Browse Job หลังจากนั้น ให้ user เลือกงานที่ต้องการ เมื่อเลือกงานที่ต้องการแล้วให้กดปุ่ม Apply

20) User interface design including relevant messages for WND of topic 18)

UI for Browse Job Use Case

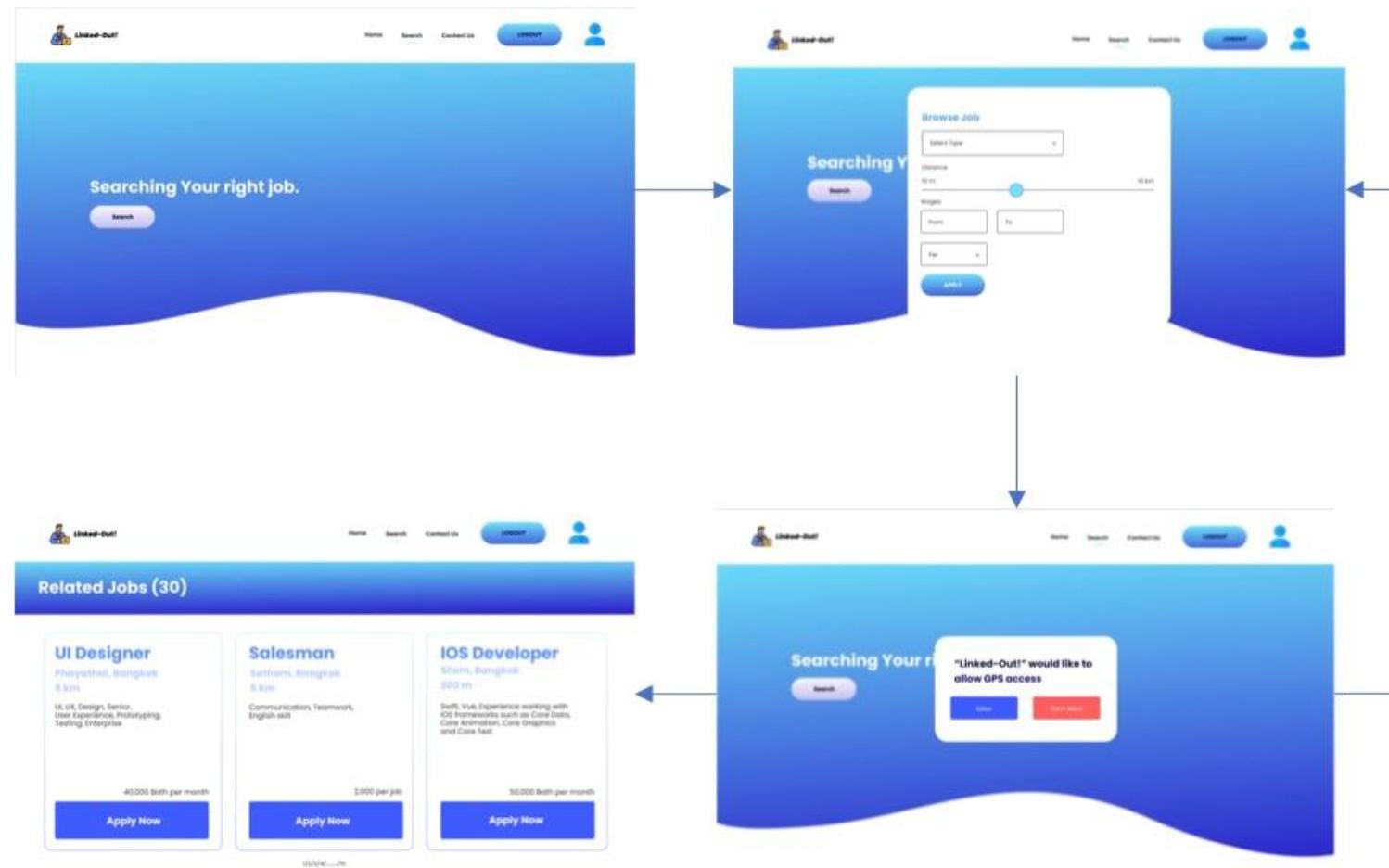


Figure 22 UI For Browse-Job Use Case

UI Design Prototype

ต้นแบบของการออกแบบส่วนต่อประสานผู้ใช้งาน เพื่อให้เห็นภาพของการทำงานมากยิ่งขึ้น

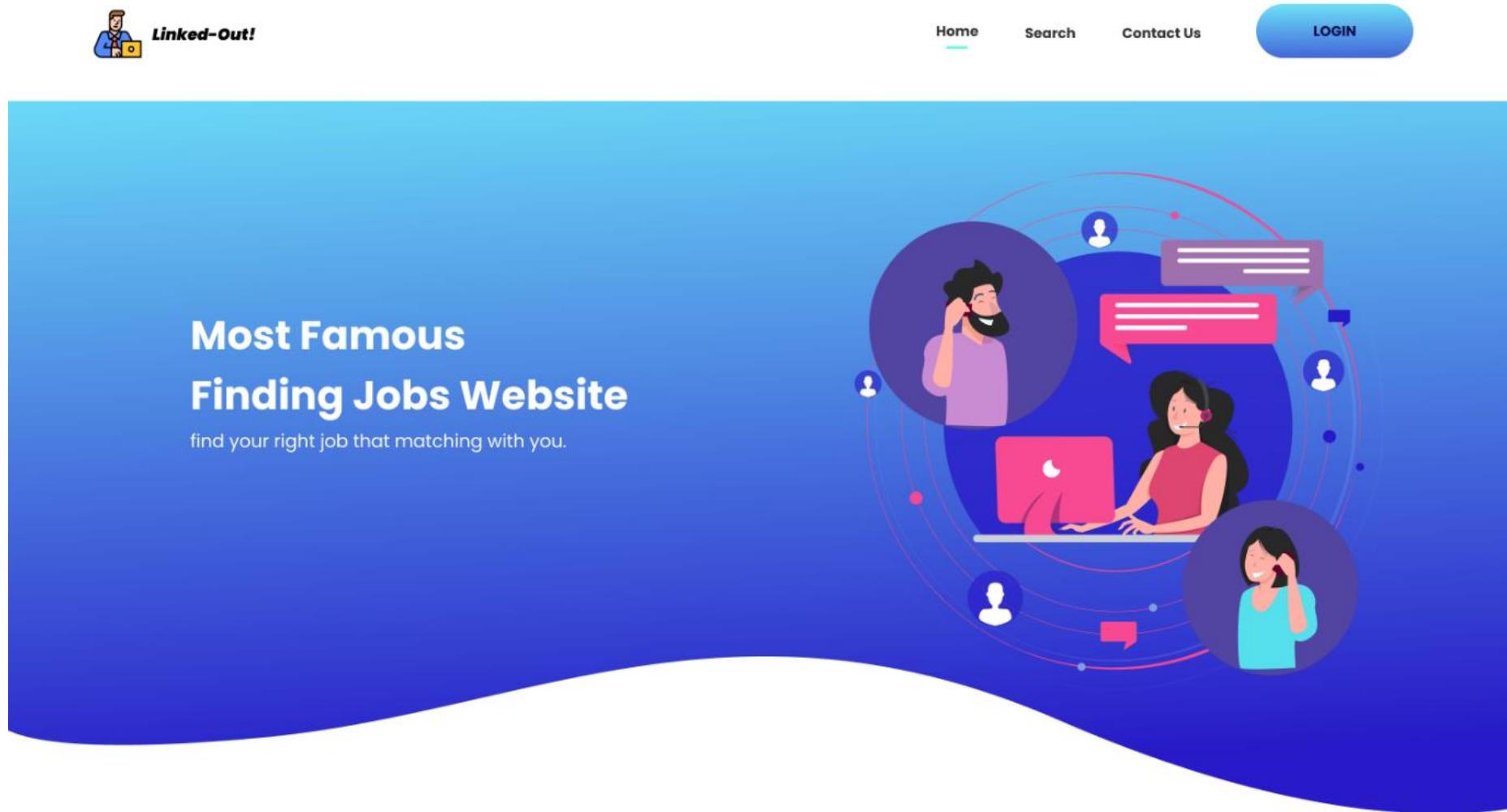


Figure 23 Home Page

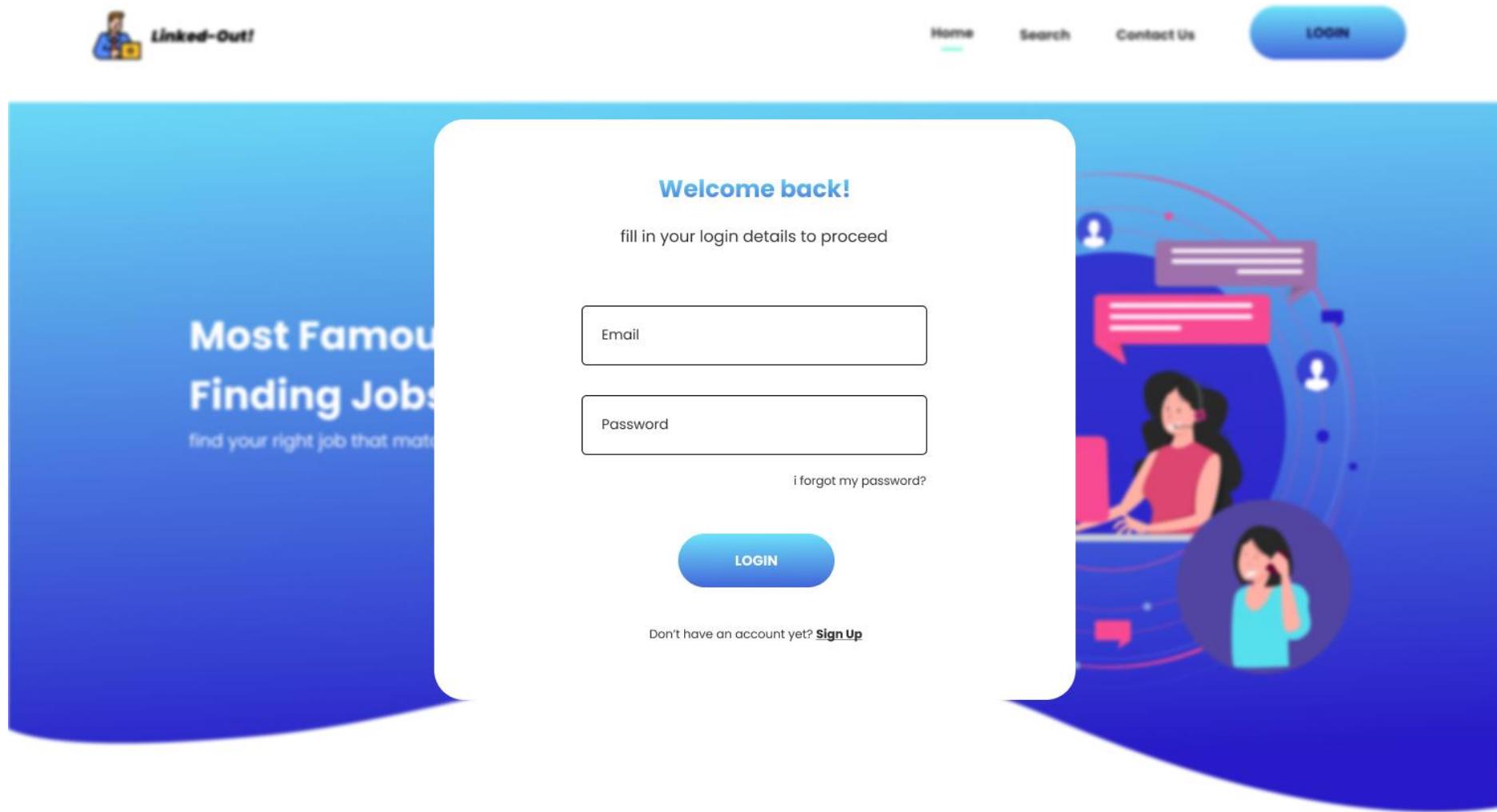


Figure 24 Login Screen

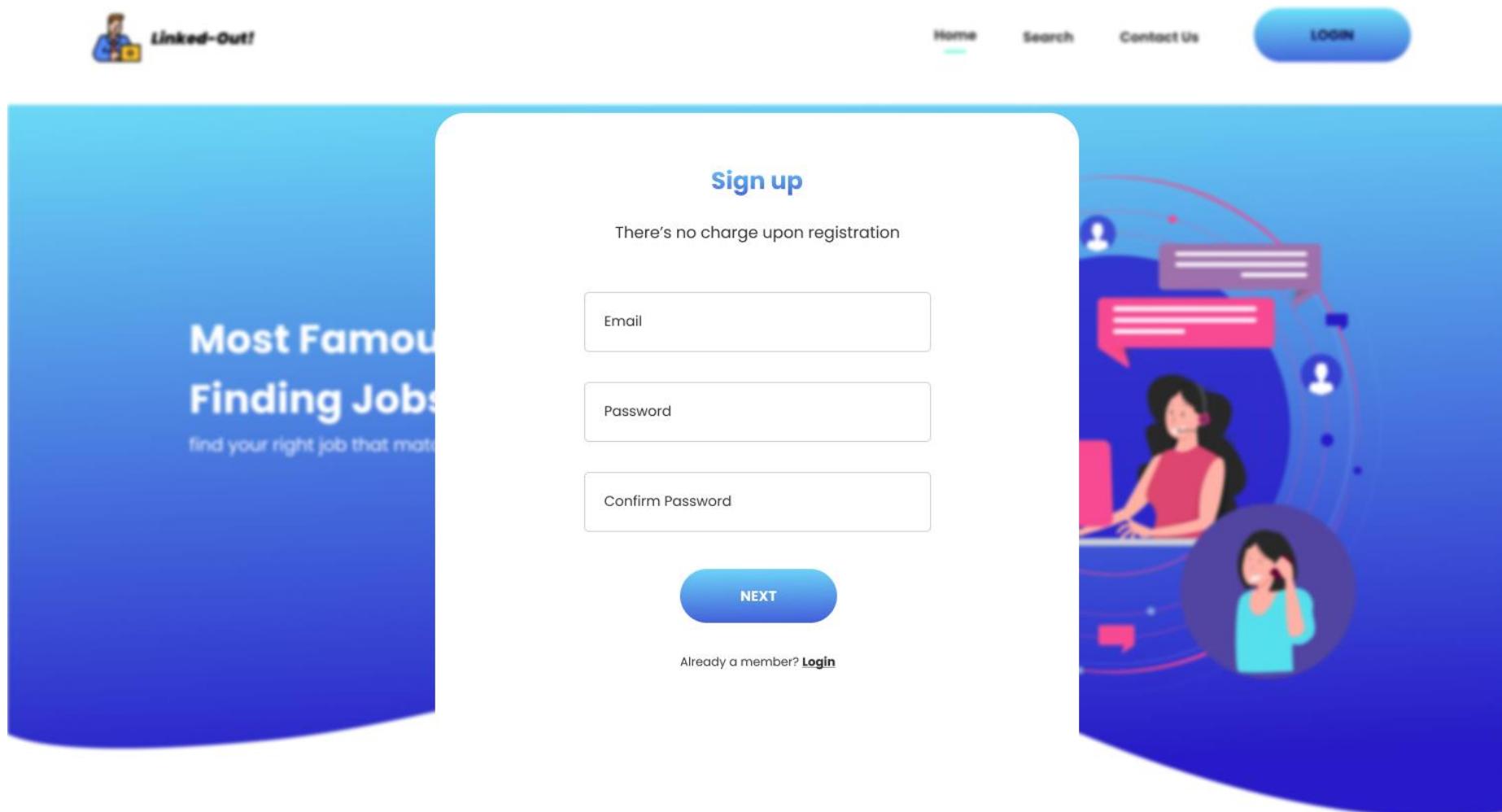


Figure 25 Sign Up Screen

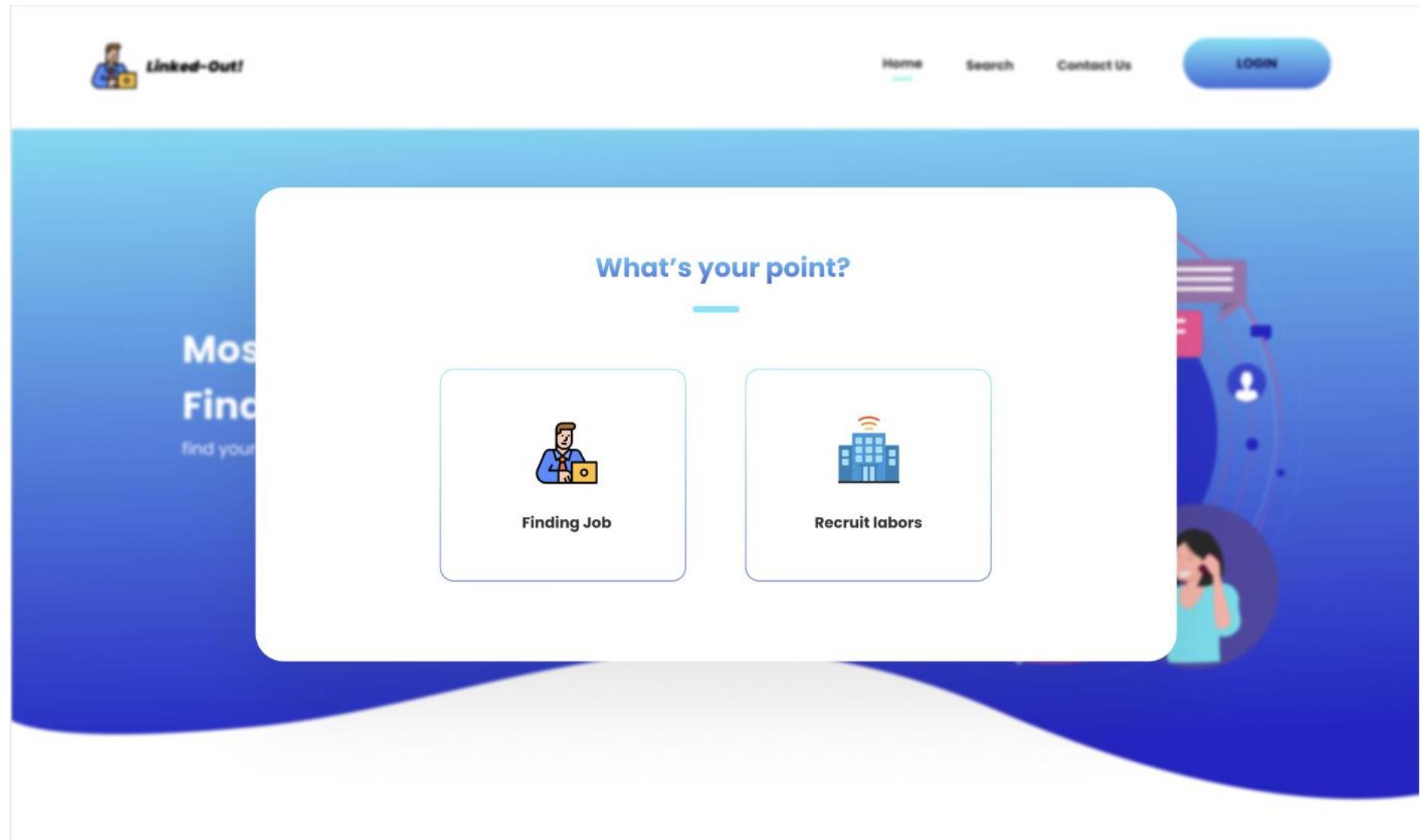


Figure 26 Sign Up Screen (2)

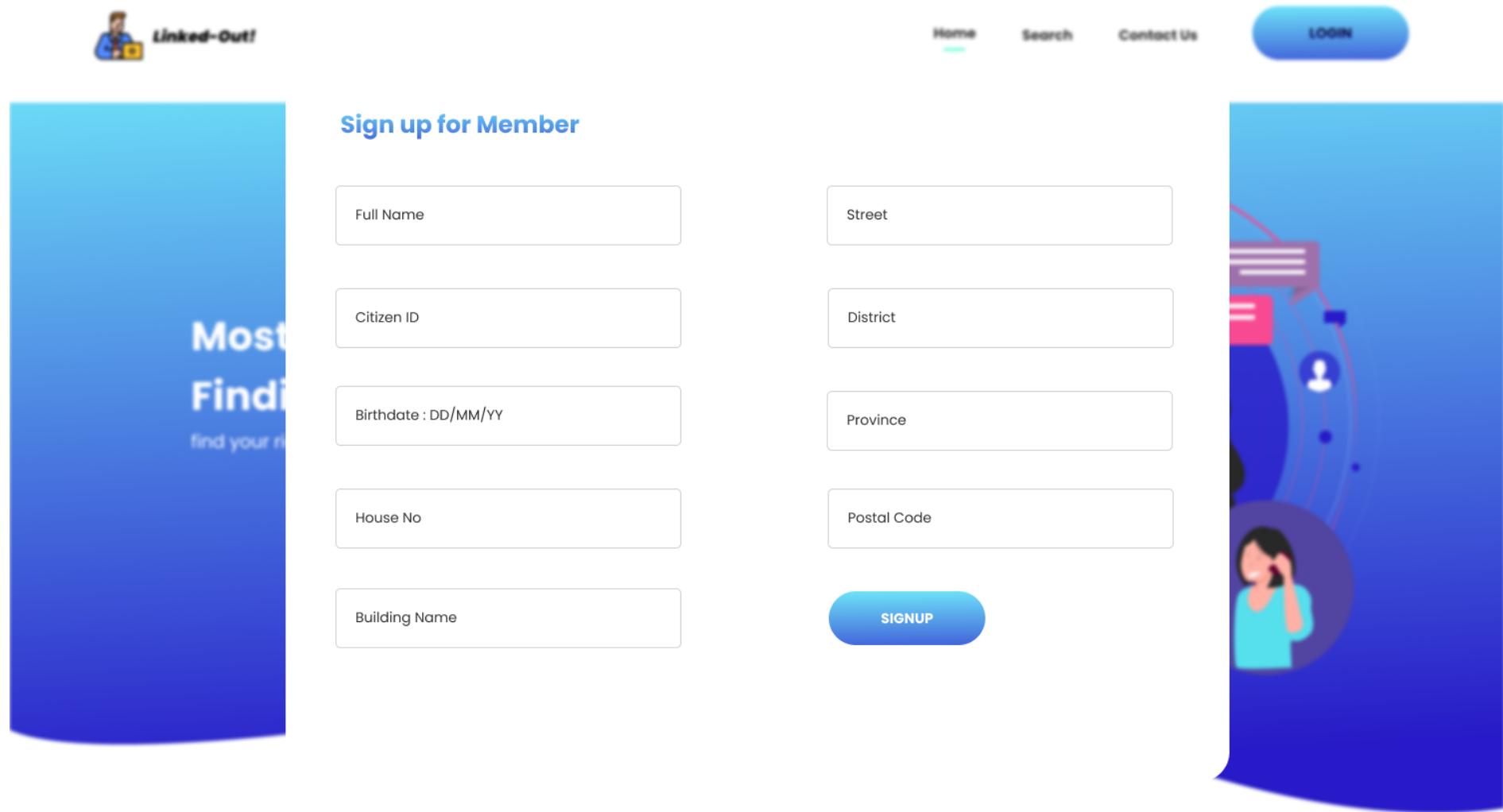


Figure 27 Sign up For Member Screen (After Clicking “Finding Job” in Figure 26)

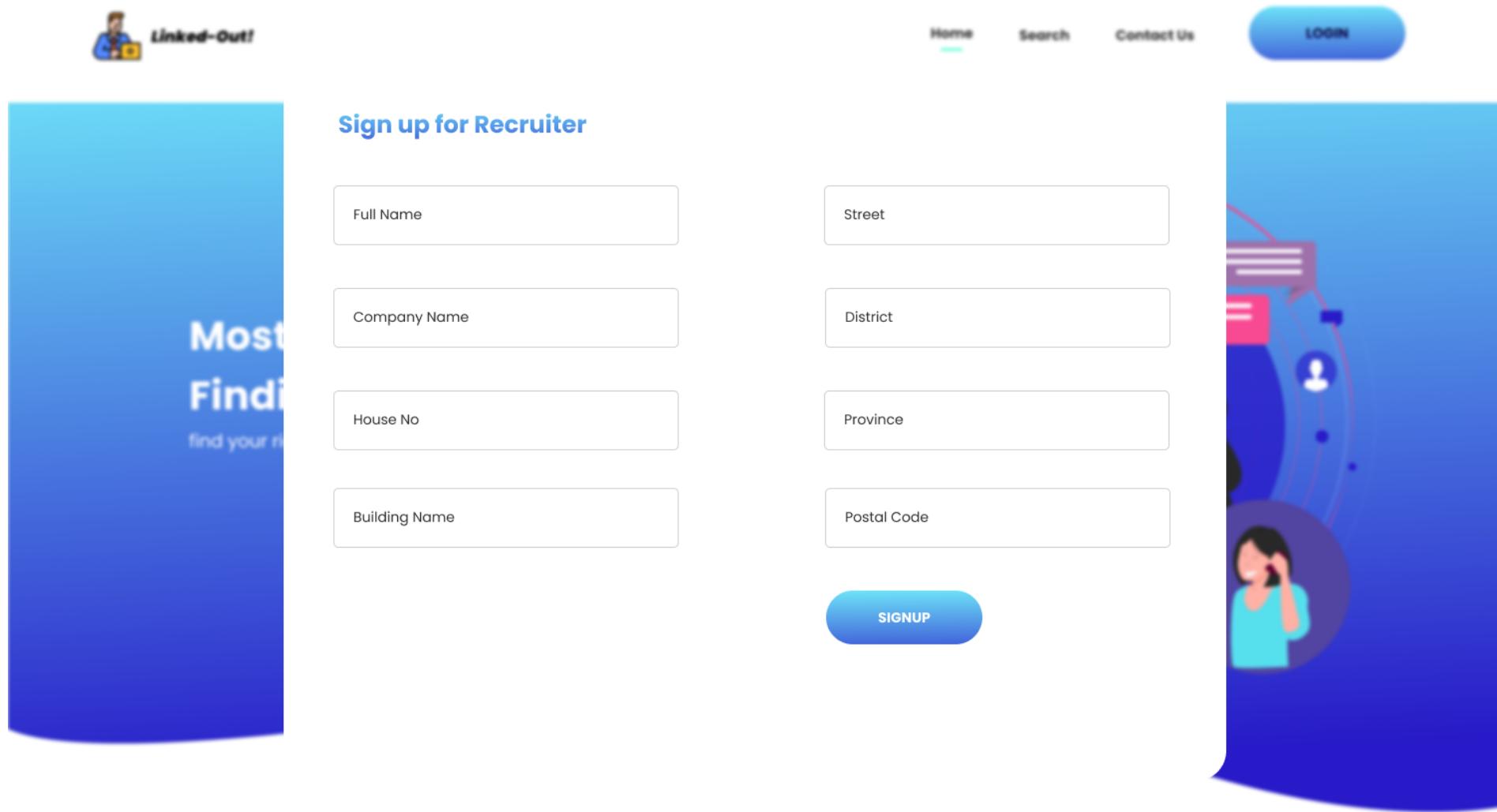


Figure 28 Sign up For Recruiter Screen (After Clicking "Recruiter Labors" in Figure 26)

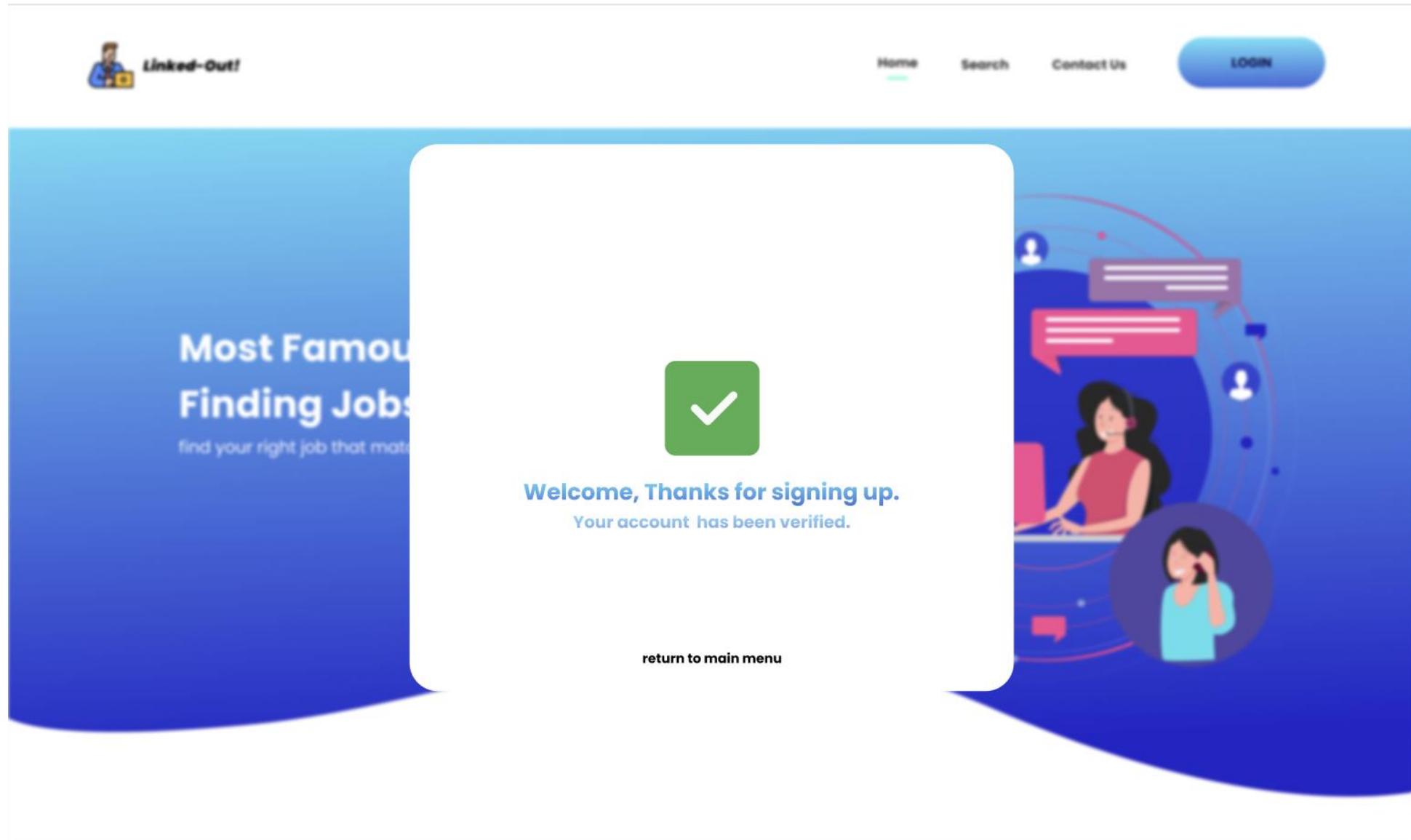


Figure 29 Success Sign Up Screen

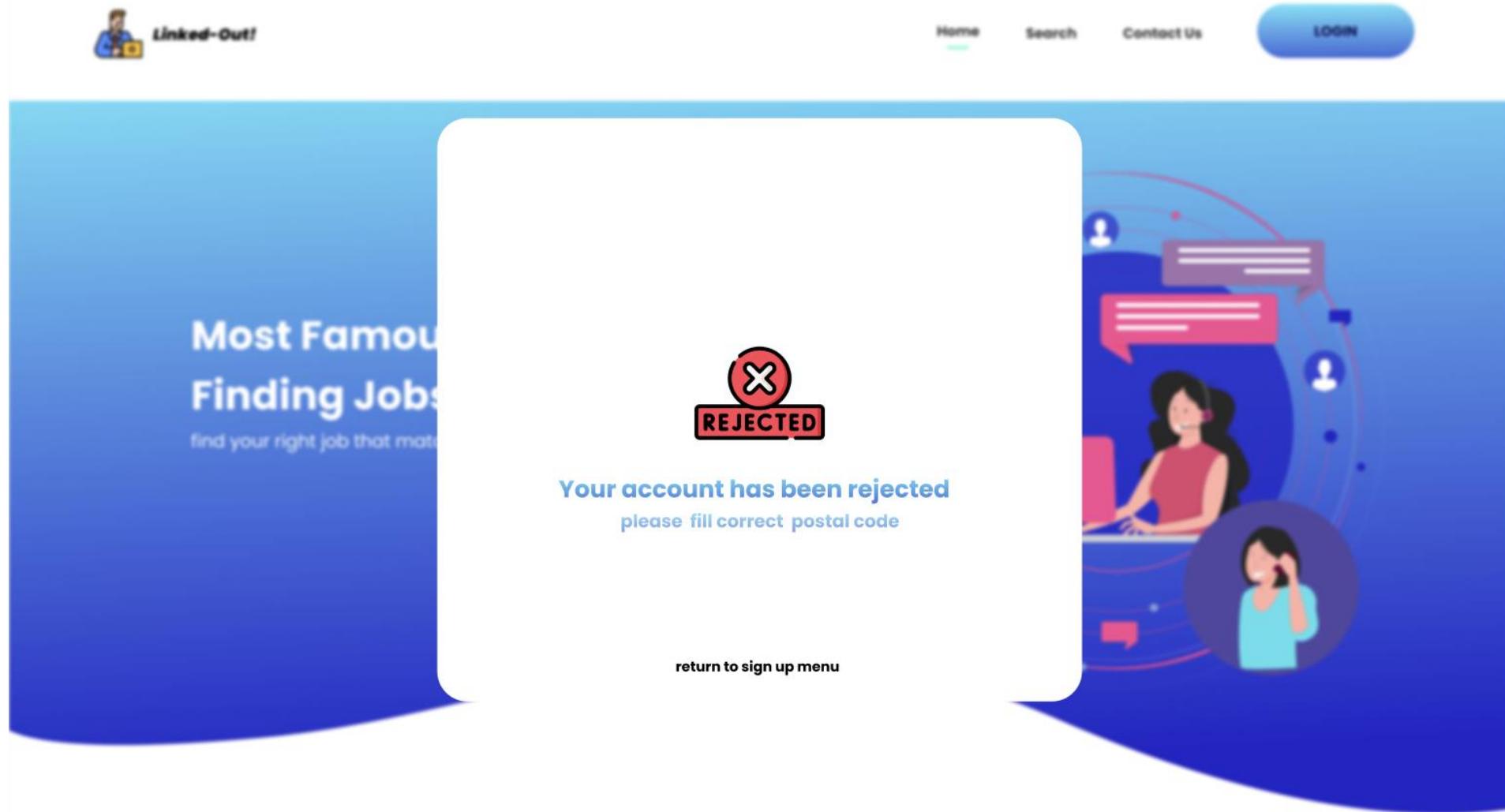


Figure 30 Fail Sign Up Screen

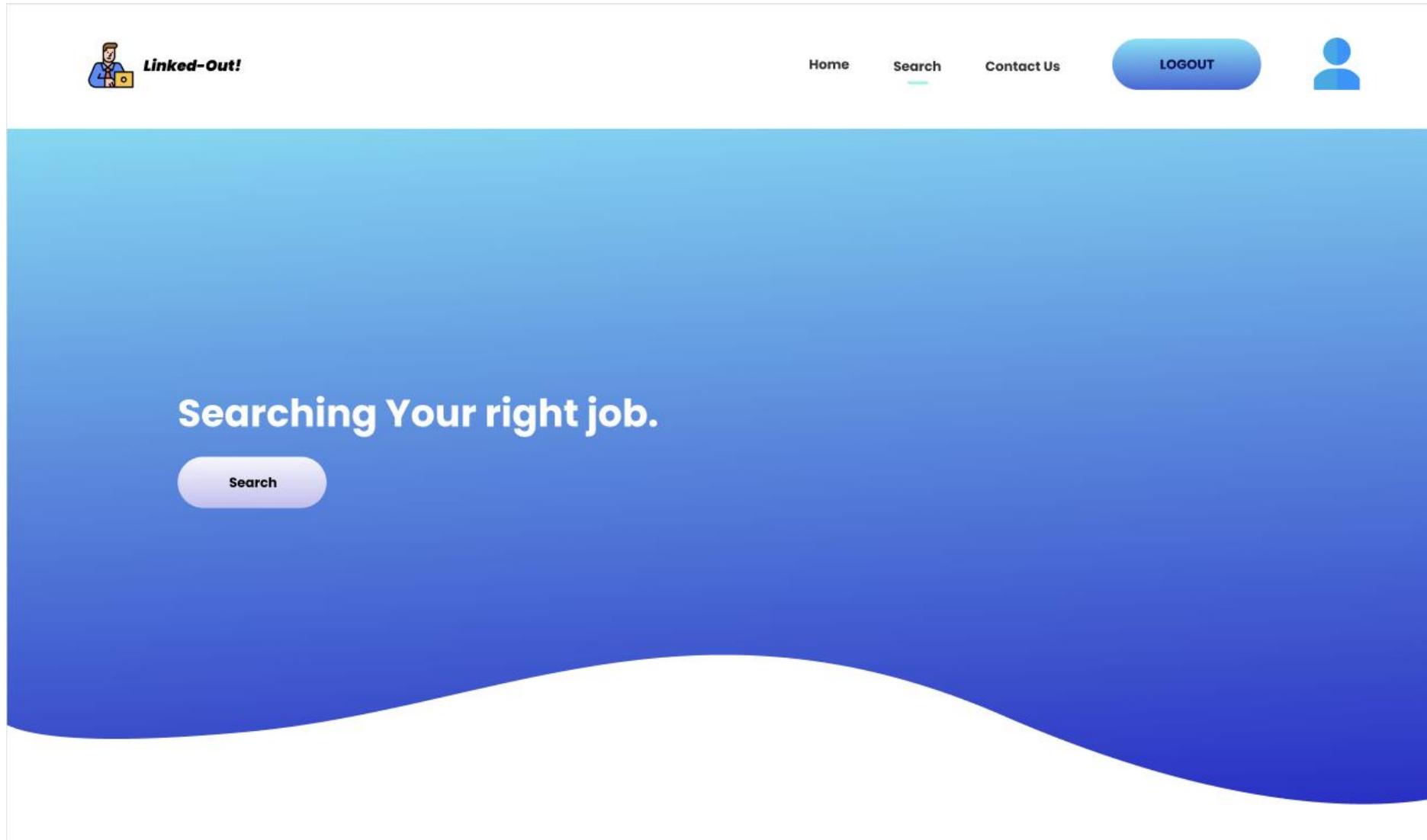


Figure 31 Search Screen

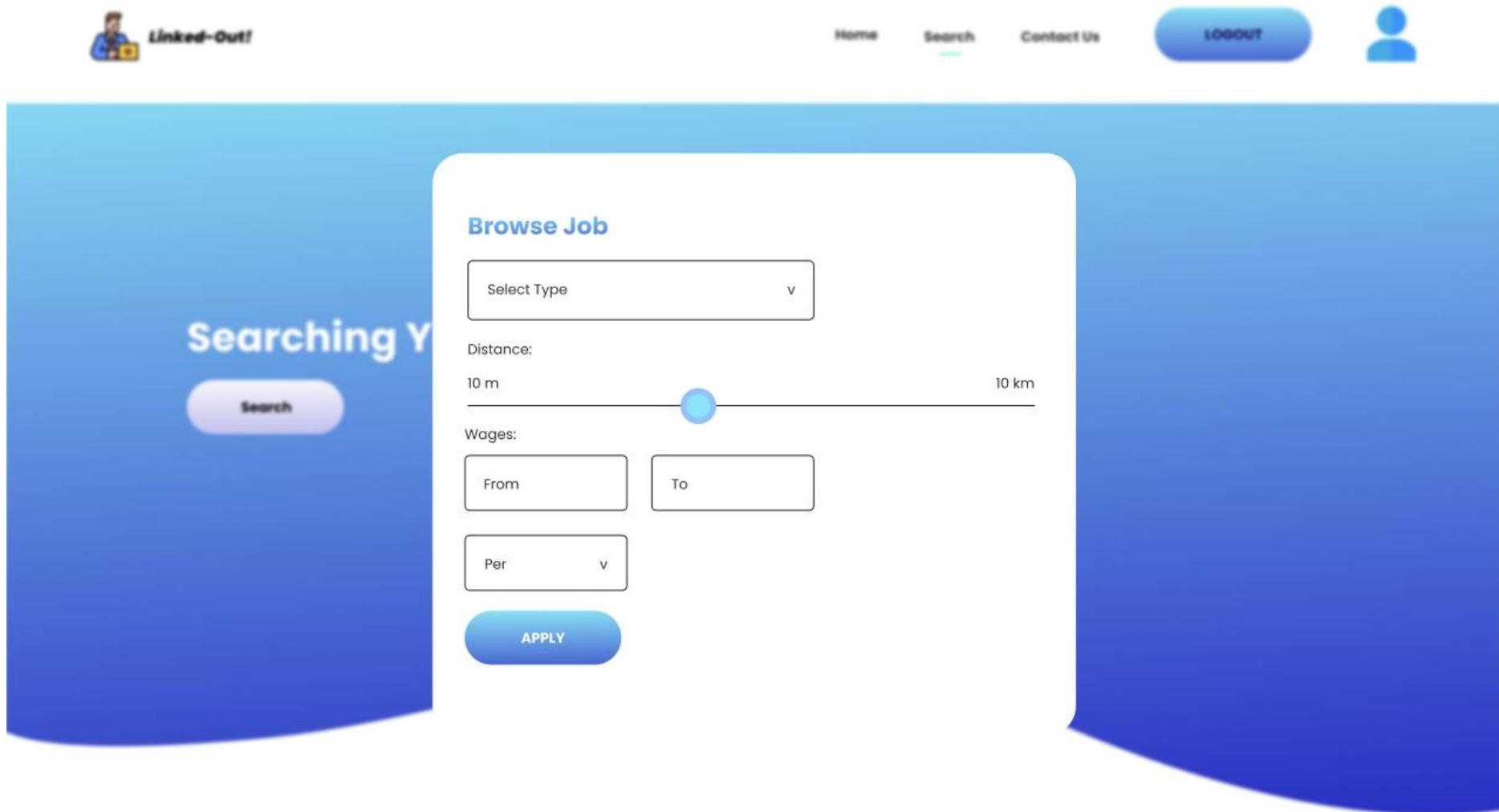


Figure 32 Search Screen (After Clicking “Search” in Figure 31)

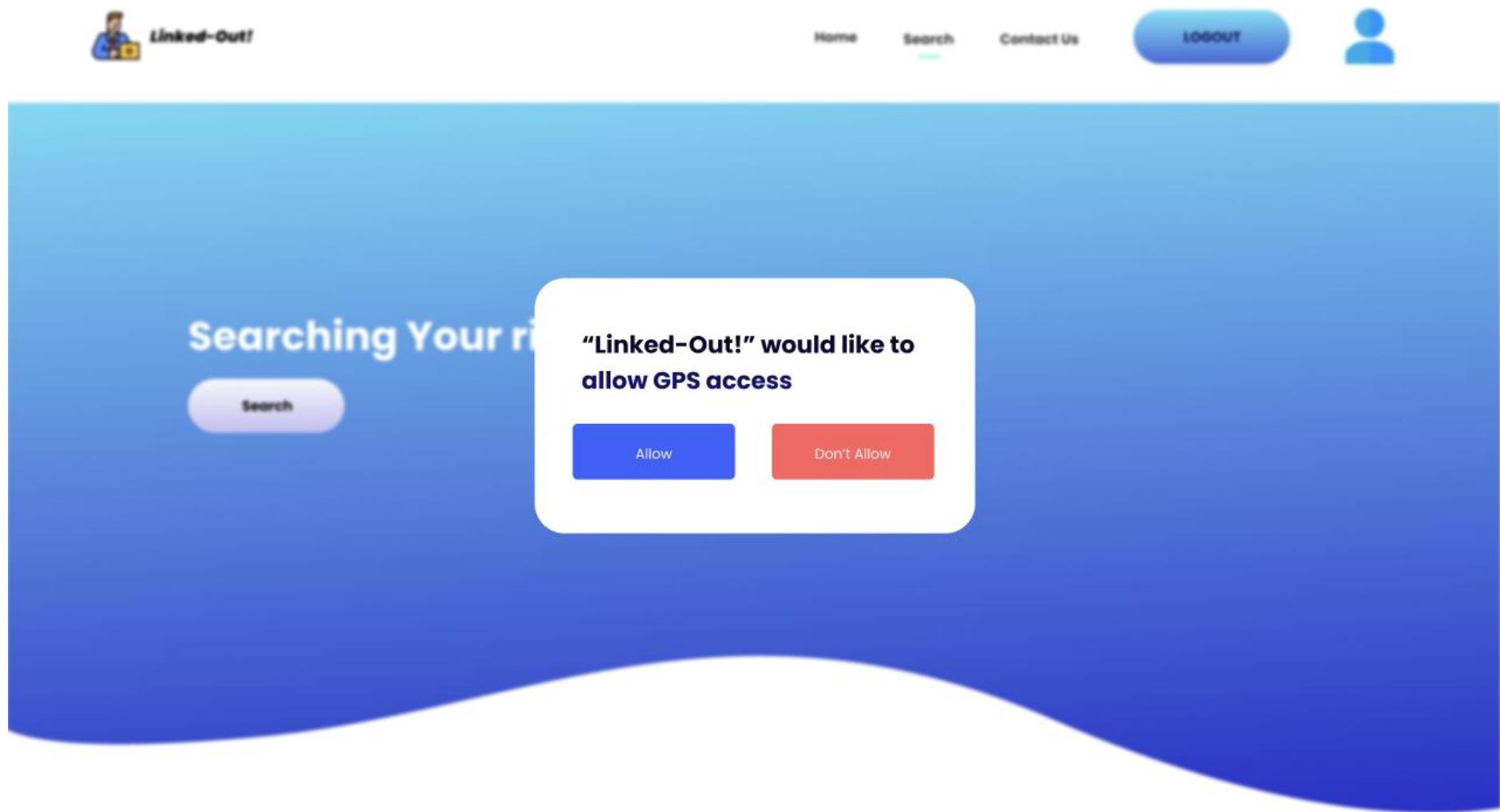


Figure 33 Requiring GPS access Screen (After Clicking “APPLY” in Figure 32)

[Home](#)[Search](#)[Contact Us](#)[LOGOUT](#)

Related Jobs (30)

UI Designer

Phayathai, Bangkok

9 km

UI, UX, Design, Senior, User Experience, Prototyping, Testing, Enterprise

40,000 Bath per month

[Apply Now](#)

Salesman

Sathorn, Bangkok

5 km

Communication, Teamwork, English skill

2,000 per job

[Apply Now](#)

IOS Developer

Silom, Bangkok

200 m

Swift, Vue, Experience working with iOS frameworks such as Core Data, Core Animation, Core Graphics and Core Text

50,000 Bath per month

[Apply Now](#)

1/2/3/4/...../10

Figure 34 Job Feeds

The screenshot shows a job application interface. At the top, there's a navigation bar with a user icon labeled "Linked-In!", "Home", "Search", "Contact Us", a blue "LOGOUT" button, and a profile icon. Below this, a blue header bar displays "Related Jobs (30)". The main content area features three job listings:

- UI Designer** in Phayathai, Bangkok, 9 km. Description: UI, UX, Design, Senior, User Experience, Prototyping, Testing, Enterprise. Salary: 40,000 Bath per month. Call-to-action: **Apply Now**.
- A central upload section titled "Upload your Résumé/CV" with options to "Drag files to here or select from device". It also has a "Uploads" button.
- UX Developer** in Bangkok, 1 km. Description: UI, Design, Senior, User Experience, Prototyping, Testing, Enterprise. Salary: 50,000 Bath per month. Call-to-action: **Apply Now**.

At the bottom, a footer bar shows page navigation: 1/2/3/4/...../10.

Figure 35 Applying job Screen (After Clicking "Apply Now" in Figure 34)

The screenshot shows a user interface for a job application platform. At the top, there is a navigation bar with links for Home, Search, Contact Us, LOGOUT, and a user profile icon. A banner at the top indicates 'Related Jobs (30)'. Below this, two job listings are shown: 'UI Designer' in Phayathai, Bangkok (9 km) and 'S Developer' in Bangkok (1 km). Both listings include a brief description of the role and an 'Apply Now' button. In the center, a green checkmark icon and the message 'Your Résumé/CV is sent Good luck' are displayed, indicating the successful submission of the resume. The bottom of the screen shows page navigation with '1/2/3/4/...../10'.

Linked-In!

Home Search Contact Us LOGOUT

Related Jobs (30)

UI Designer
Phayathai, Bangkok
9 km

UI, UX, Design, Senior, User Experience, Prototyping, Testing, Enterprise

40,000 Bath per m

Apply Now

S Developer
im, Bangkok
1 km

K, Design, Senior, Experience, Prototyping, Testing, Enterprise

50,000 Bath per month

Apply Now

return to main menu

1/2/3/4/...../10

Figure 36 Applying Success Screen

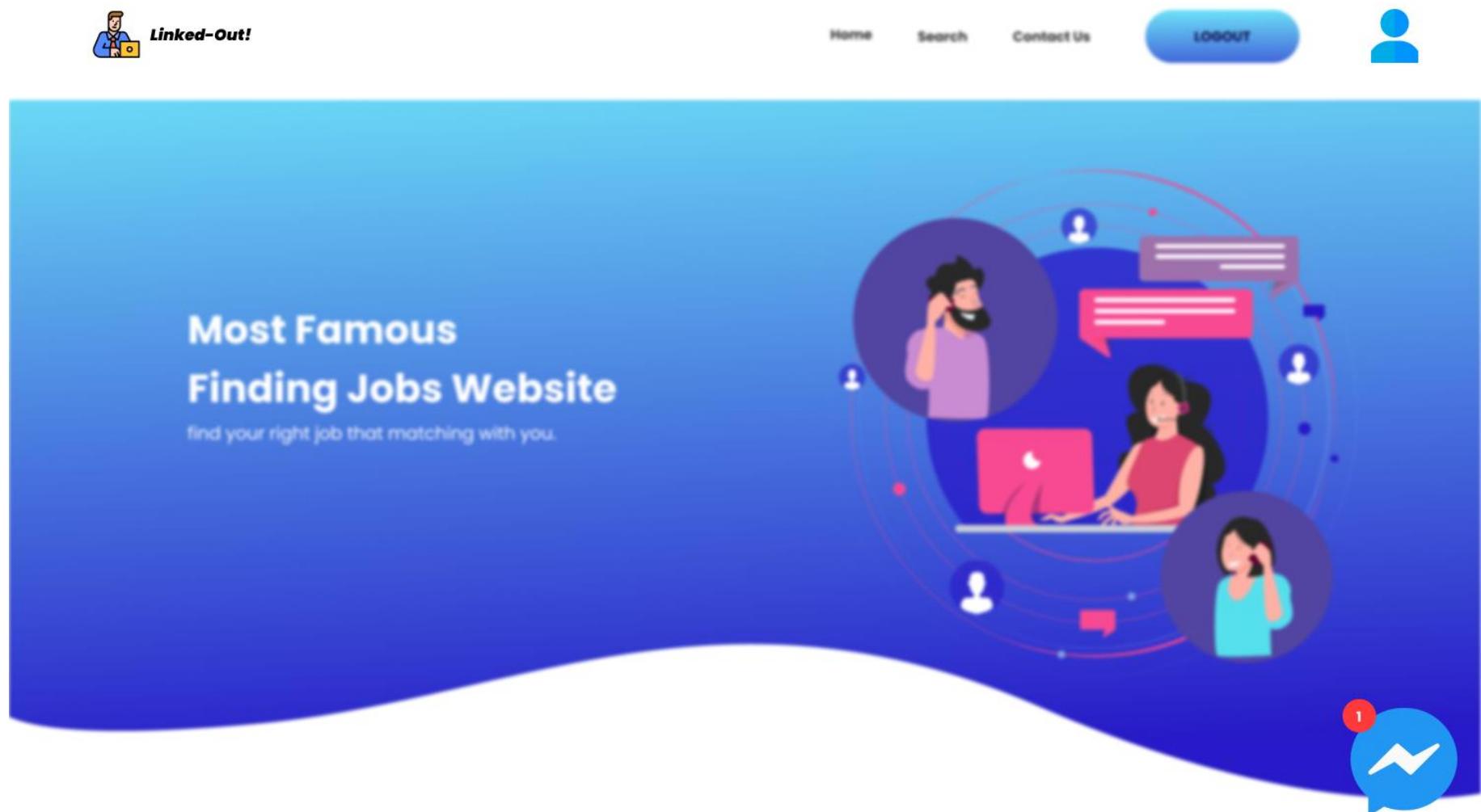


Figure 37 Chat Notification

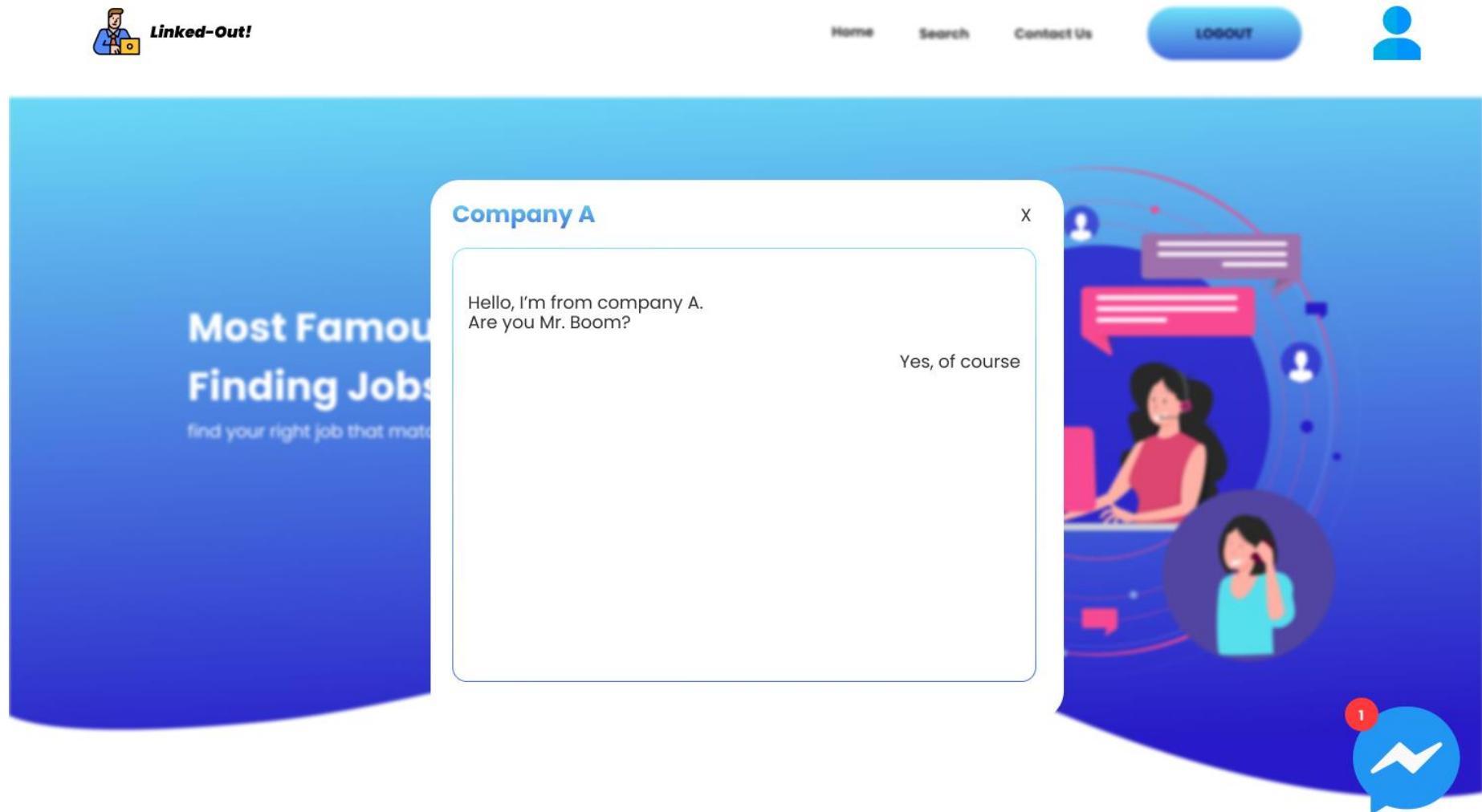


Figure 38 Chatroom Screen

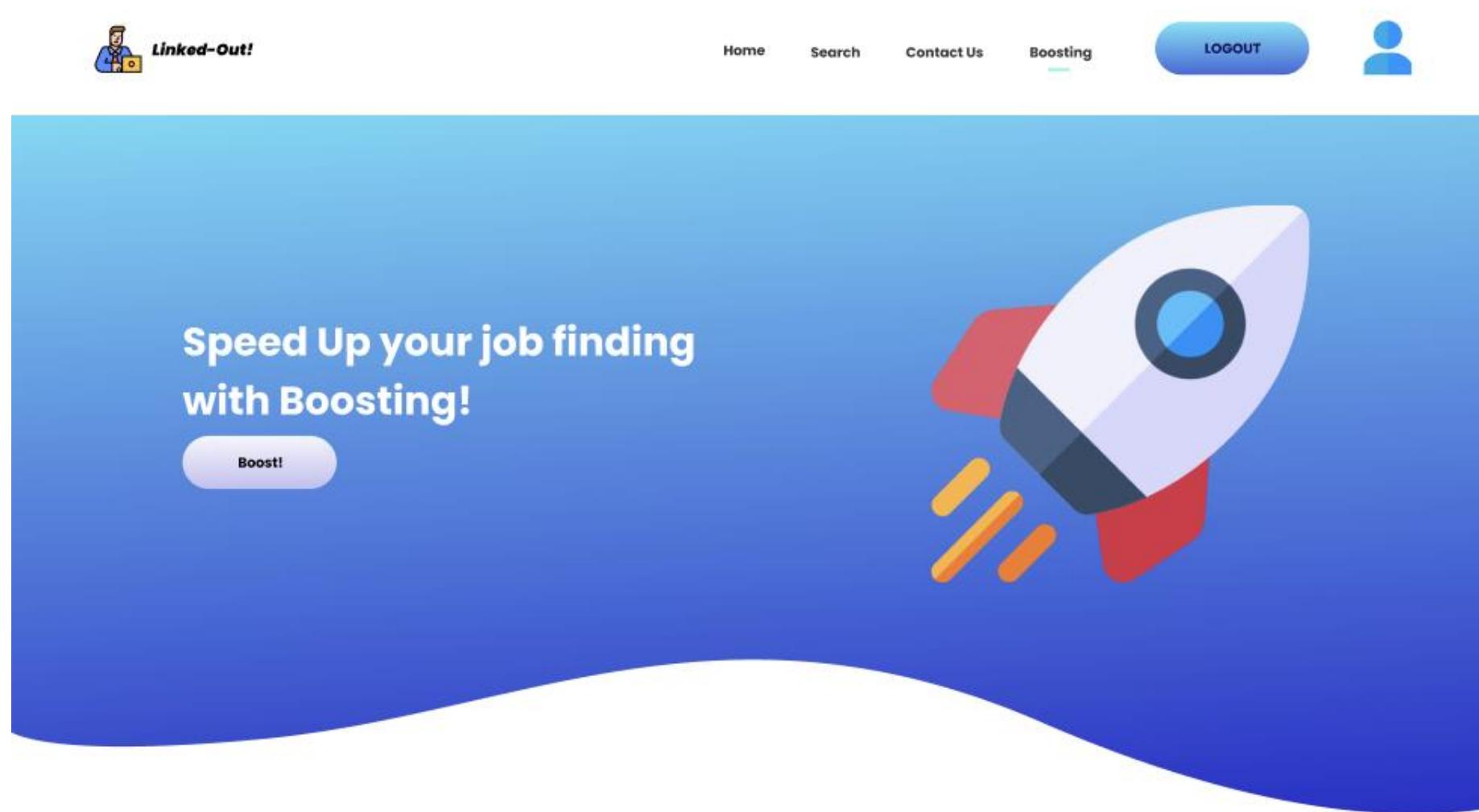


Figure 39 Boosting Screen



Figure 40 Boosting Screen (2)

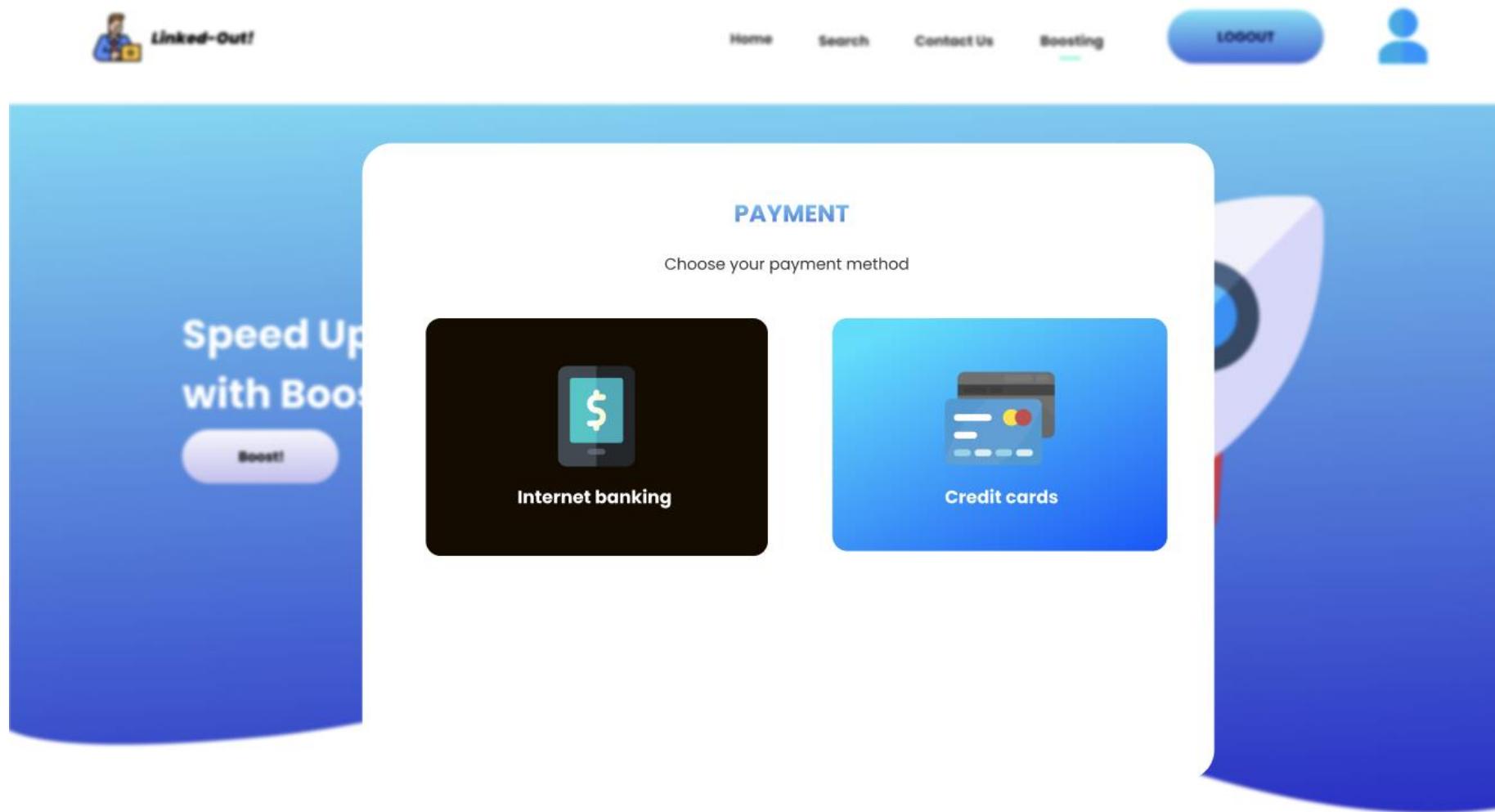


Figure 41 Payment Screen

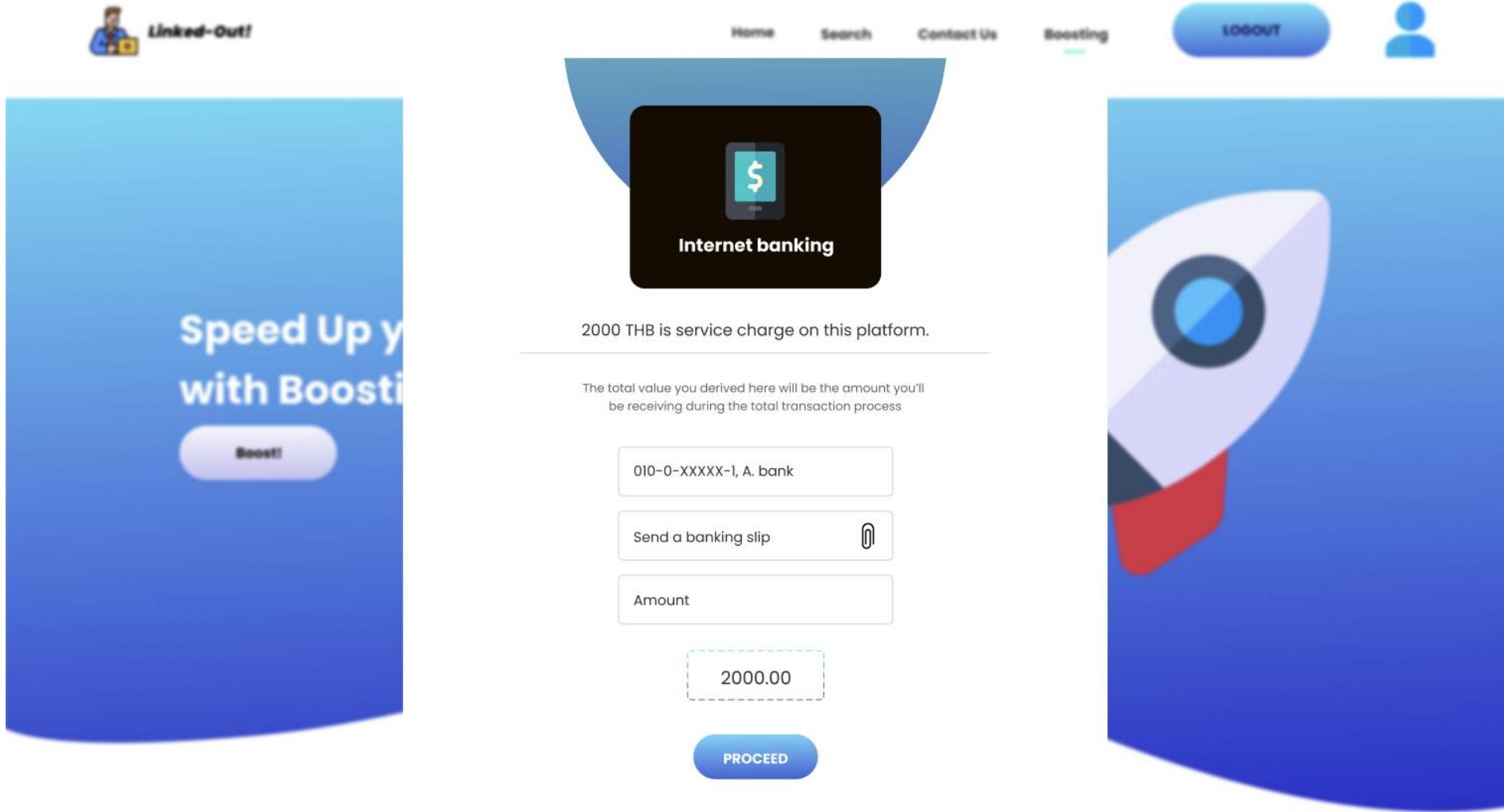


Figure 42 Pay by internet banking

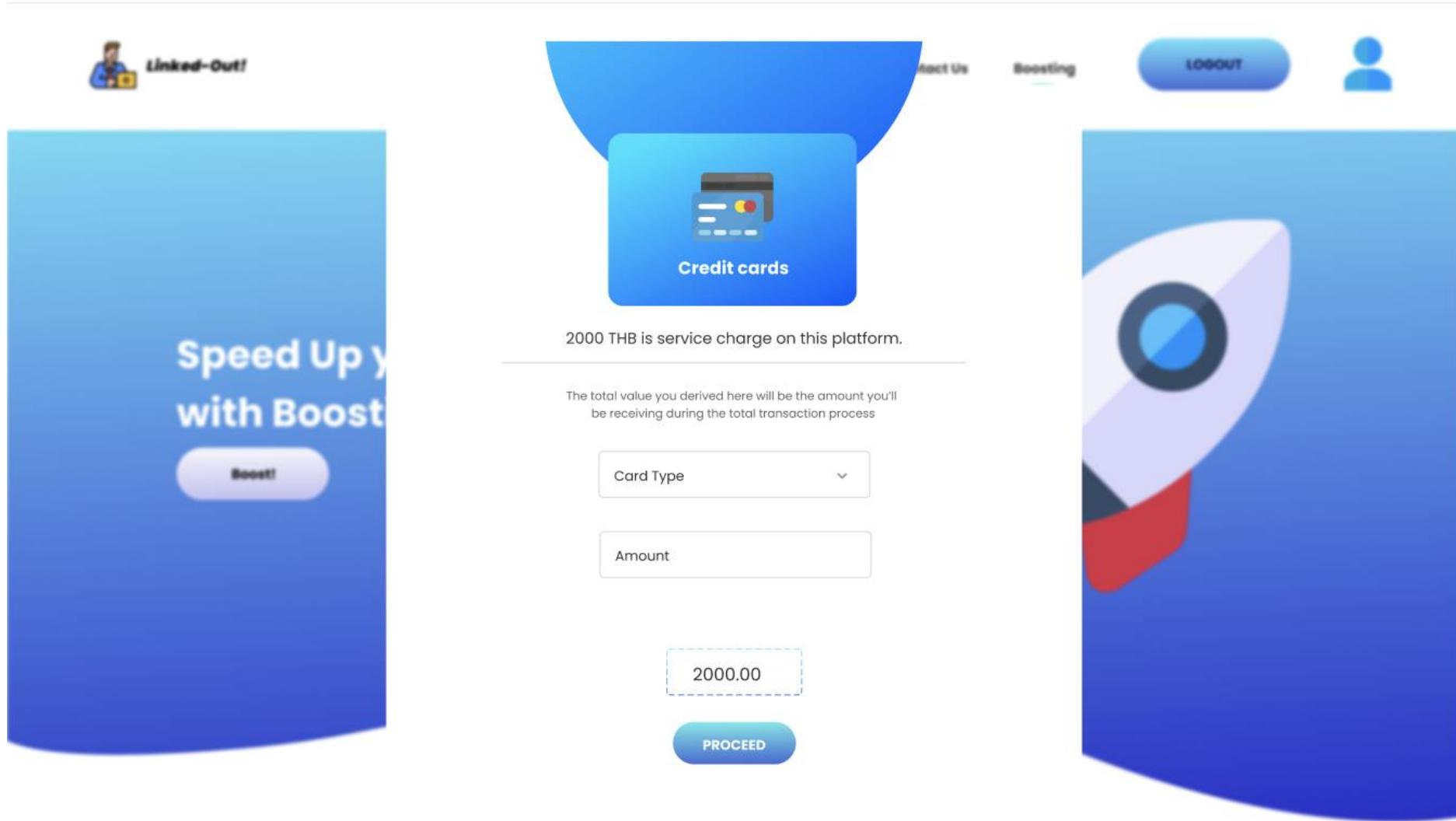


Figure 43 Pay by credit card

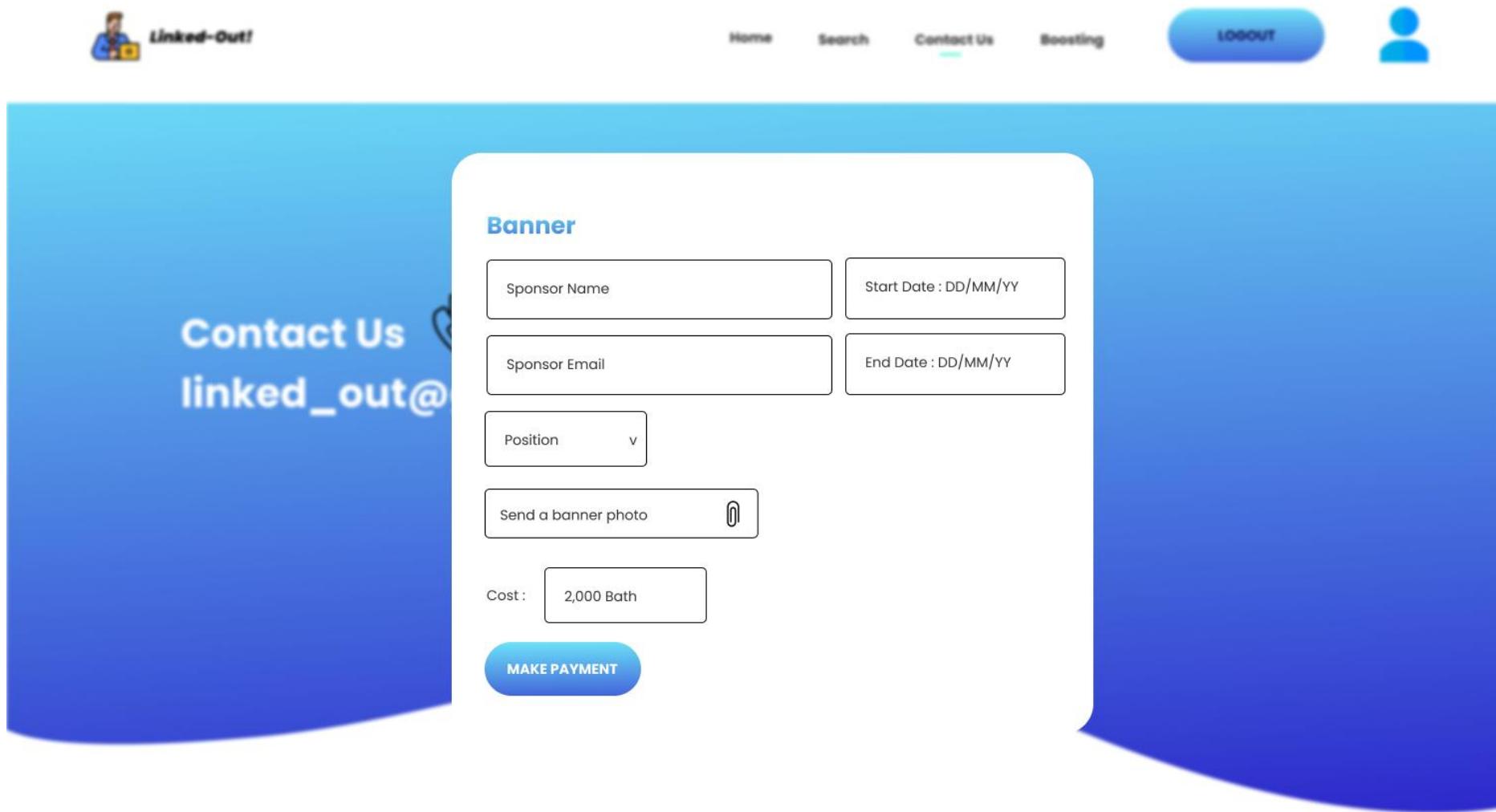


Figure 44 Buying Banner Screen

21) Nonfunctional requirements and their effect on physical architecture design

Type of Nonfunctional Requirement	Requirement	Effect for designing architectural model
Operational Requirement	ระบบต้องสามารถใช้งานได้บน โทรศัพท์มือถือและแท็บเล็ตผ่านเว็บ เบราว์เซอร์ Mozilla Firefox, Google Chrome (Android และ PC), Microsoft Edge, Safari (iOS และ MacOs) เป็นอย่างน้อยโดยต้องทำงานเหมือนกันทุกประการ (identical)	เนื่องจาก client มี specification ที่ไม่แน่นอน (เช่น ไม่สามารถกำหนด CPU หรือ RAM ของ client ได้) จึงเลือกใช้แบบ Client-Server โดยเป็นทางเลือกปกติของการทำ Web Application อยู่แล้ว โดยแบ่งเป็น 3 tiers ขึ้นไป ได้แก่ GUI, Backend logic, Data store
Operational Requirement	ระบบต้องทำสำเนาข้อมูลที่สามารถกู้คืนได้ภายในเวลา 2 ชั่วโมง (ข้อมูลสำคัญ มีการเรียกใช้งานบ่อย) โดยสำรองข้อมูล 1 ชุดทุก ๆ 1 วัน	Replicate Database อย่างน้อย 1 ชุด (Main, Secondary) โดยไม่สร้าง inconsistency (ไม่เลือกใช้ Database ที่เป็น AP Database ตาม CAP Theorem) และทำการสร้าง Snapshot ทุก 1 วัน ไปเก็บไว้ใน Data store server ที่ใช้ HDD ในการเก็บเพื่อต้นทุนที่ถูกกว่า
Operational Requirement	ระบบต้องทำสำเนาข้อมูลที่สามารถกู้คืนได้ภายในเวลา 72 ชั่วโมง (ข้อมูลไม่สำคัญ มีการเรียกใช้งานไม่บ่อยครั้ง) โดยสำรองข้อมูล 1 ชุดทุก ๆ 1 สัปดาห์	ใช้ Data store server แบบเดียวกัน โดยอาจเพิ่ม infrastructure ที่เป็น tape-based storage เพื่อระบายน้ำข้อมูลออกจาก Data store server ทุก ๆ 1 ปี
Performance Requirement	ระบบต้องสามารถใช้งานได้อย่างน้อย 22 ชั่วโมงต่อวัน	กระจาย database และ server ไปตาม availability zone 2 zone ได้แก่ asia-southeast-1a และ asia-southeast-1b (ใช้ cloud-based infrastructure เพื่อลดต้นทุนการตั้ง server)
Performance Requirement	ระบบต้องตอบสนองการใช้งานของบุคคลใด ๆ ได้ภายในระยะเวลา 1 วินาที	ทำตัว autoscale backend และแยก database ออกมาเป็น read/write databases เพื่อความเร็วในการอ่าน
Performance Requirement	ระบบต้องรองรับอัตราการเข้ามาร่องต่อจากผู้ใช้ได้อย่างน้อย 300 คนต่อชั่วโมง	สร้าง replicate ใน backend และ Database โดยใช้ load-balancer เป็น gateway ในการเข้าม至 service

Security Requirement	ระบบต้องใช้พอร์ต HTTPS ในการรับ-ส่งข้อมูลใดๆจากบุคคลใด ๆ ที่เข้ามาใช้งานระบบ	ทำ central routing server ซึ่งจะทำหน้าที่เป็นทั้ง load balancer และ traffic router ซึ่งจะคอยดูแล traffic ทั้งหมดในฝั่ง backend และ database โดย traffic ใดที่ต้องใช้ HTTPS ก็ให้ทำการ setup TLS ที่ขั้นตอนนี้
----------------------	--	---

Table 9 Nonfunctional requirements and their effect on physical architecture design

22) Overview of infrastructure design

Diagram ดังต่อไปนี้แสดงเครือข่ายของ Application ของเรา ซึ่งจะ deploy ใน Google Kubernetes Engine ในส่วนของ Backend และ Database โดยจะเชื่อมต่อกับส่วน Frontend ซึ่งอยู่ใน Web Browser ของผู้ใช้งานผ่าน Internet ส่วนระบบ Backend กระจายไปใน 2 Availability zone เพื่อให้ Application ของเราพร้อมใช้งานตลอดเวลา โดยใช้ Load Balancer เป็นตัวตัดสินใจแบ่ง traffic และเป็น Router ไปในตัว Database ก็แบ่งตาม Availability zone เช่นกันโดยมีการใช้ Autoscaler เพื่อ optimize จำนวน Pod

ส่วนการ Backup Database ให้ใช้ Scheduled snapshot ซึ่งเป็น feature ของ GCP ที่สามารถใช้งานได้ โดยเราจะทำเป็นรายวัน รายสัปดาห์ตาม requirement และอาจเพิ่มรายปี โดยใช้ข้อมูลจาก snapshot รายวันและรายสัปดาห์เพื่อลดรายจ่ายที่ต้องเก็บ data ที่ไม่ได้เรียกใช้บ่อยมากนัก (รูปแบบการเก็บรายปีจะใช้การเก็บข้อมูล cold storage ที่ต้นทุนน้อยกว่า)

Network Diagram

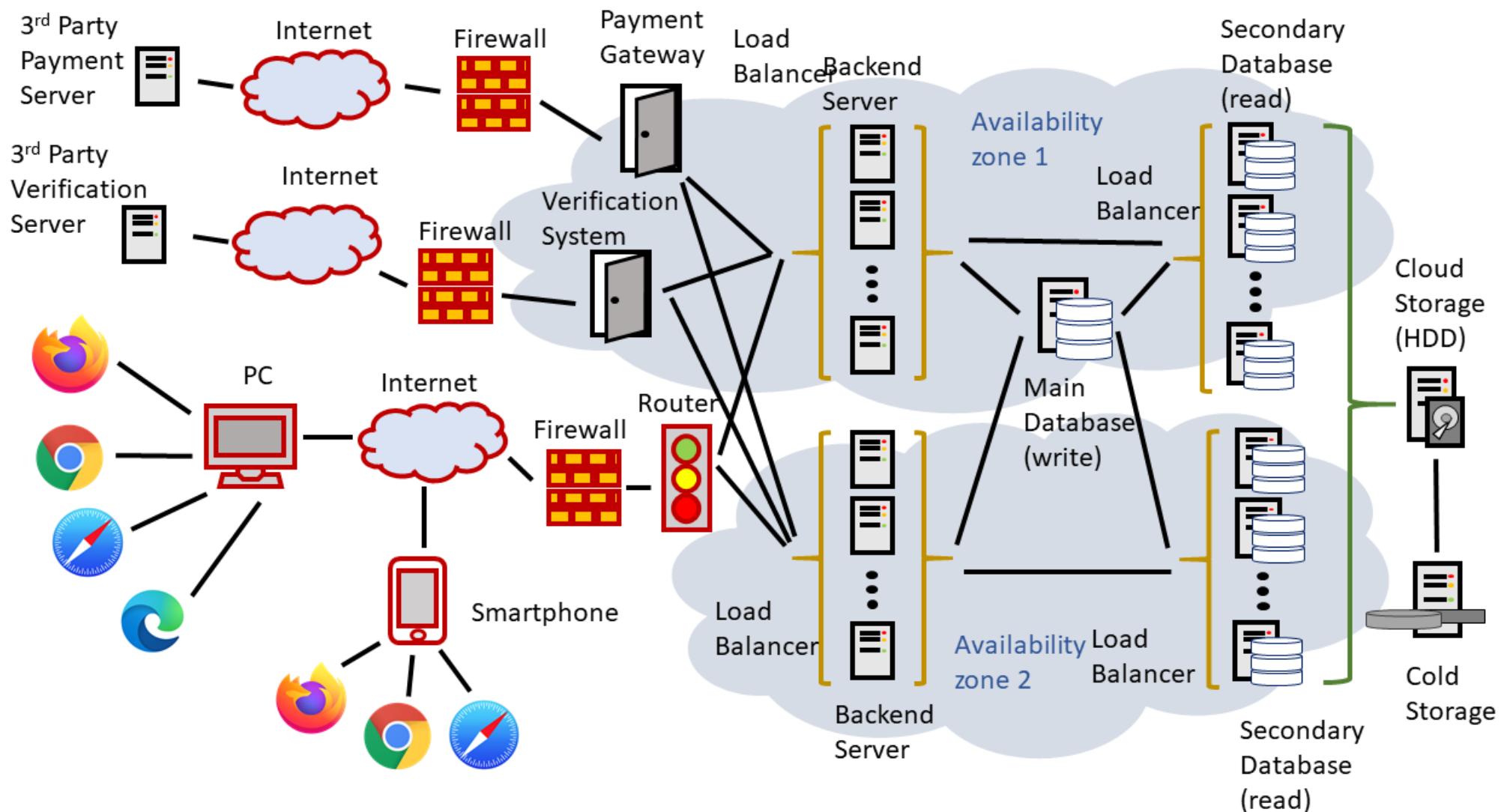


Figure 45 Network Diagram

Deployment Diagram

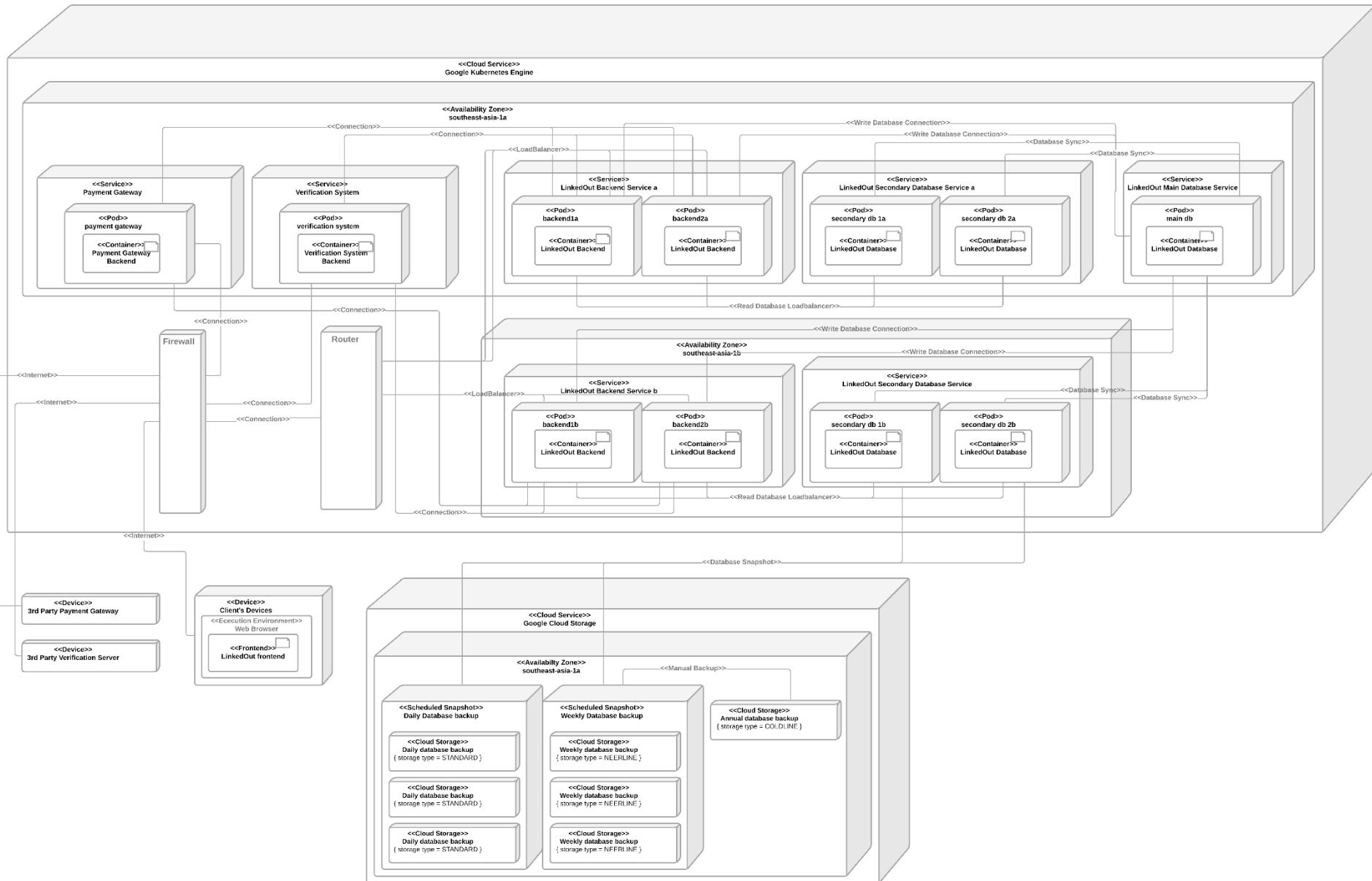


Figure 46 Deployment Diagram

สำหรับรายละเอียดทางด้าน Architecture ตามข้างต้น จะเห็นว่ามีการสร้าง replicate ของทั้ง backend และ database จำนวนมาก รวมถึงมีการกระจาย server ไปตาม availability zone ต่าง ๆ ซึ่งอาจทำให้ต้นทุนของการ deploy มากเกินกว่าที่เคยประเมินไว้ใน Feasibility study ทั้งนี้เพื่อเป็นการลดต้นทุนของการ deploy ให้เป็นไปตามการคาดการณ์ และยังคงคุณสมบัติตาม non-functional requirement จึงได้มีการออกแบบ Architecture รูปแบบที่ 2 ซึ่งจะต้องอ้างสมมติฐานบางประการเพื่อให้สามารถขยาย non-functional requirement ได้

ทั้งนี้ เนื่องจากระบบของเราใช้ Google Cloud Platform ซึ่งเป็น IaaS (Infrastructure as a Service) เมื่อ Application ของเรา มีผู้ใช้งานมากขึ้น ก็สามารถวางแผนเพื่อ Migrate ไปใช้ระบบที่ HA ได้โดยง่าย

Low-cost Deployment Modification

Type of Nonfunctional Requirement	Requirement	Modification
Operational Requirement	ระบบต้องทำสำเนาข้อมูลที่สามารถกู้คืนได้ภายในเวลา 2 ชั่วโมง (ข้อมูลสำคัญ มีการเรียกใช้งานบ่อย) โดยสำรองข้อมูล 1 ชุดทุก ๆ 1 วัน	สร้าง Read Database 1 instance และ Write Database 1 instance โดยใช้ Nearline Storage ของ Google Cloud Platform เป็นประเภทของการเก็บของมูลโดย assume ว่าภายใน 1 เดือน จะเกิดข้อมูลเสียหายอันเป็นเหตุให้ต้องใช้ backup ไม่เกิน 1 ครั้ง
Operational Requirement	ระบบต้องทำสำเนาข้อมูลที่สามารถกู้คืนได้ภายในเวลา 72 ชั่วโมง (ข้อมูลไม่สำคัญ มีการเรียกใช้งานไม่บ่อยครั้ง) โดยสำรองข้อมูล 1 ชุดทุก ๆ 1 สัปดาห์	ไม่ทำสำรองข้อมูลรูปแบบนี้ เนื่องจากมีความซ้ำซ้อนกับการสำรองของข้อมูลรายวัน โดยทำการสำรองรายวันทำทั้ง database แทนการทำเพียงบางส่วน แต่ยังคงสำรองข้อมูลรายปีเหมือนเดิม แต่เปลี่ยนไปใช้ archive storage แทน โดย assume ว่าภายใน 1 ปี จะเกิดข้อมูลเสียหายอันเป็นเหตุให้ต้องใช้ backup นี้ไม่เกิน 1 ครั้ง
Performance Requirement	ระบบต้องสามารถใช้งานได้อย่างน้อย 22 ชั่วโมงต่อวัน	ให้ทั้ง database และ backend อยู่ใน availability zone เดียว คือ asia-southeast-1a โดยมี backend ที่ 1 replica ซึ่งสามารถ scale ได้อย่างอัตโนมัติ (Assume ว่าไม่มี traffic spike เพื่อให้ app สามารถ scale อย่างอัตโนมัติโดยไม่ติดคอกขาดที่ autoscaler)

Low-cost Network Diagram

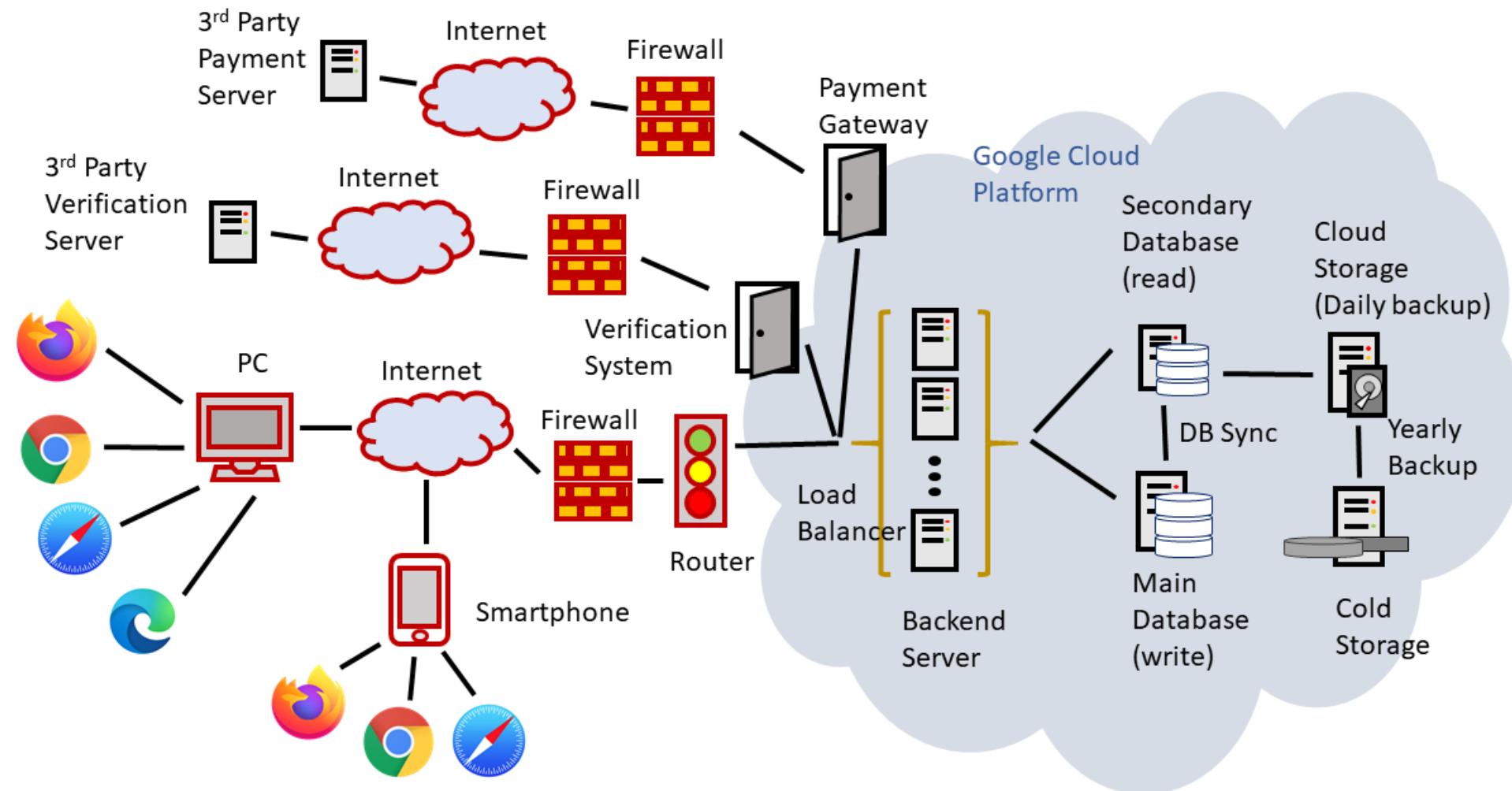


Figure 47 Low-cost Network Diagram

Figure 48 Low-cost Deployment Diagram

Figure 47 Low-cost Network Diagram

Low-cost Deployment Diagram

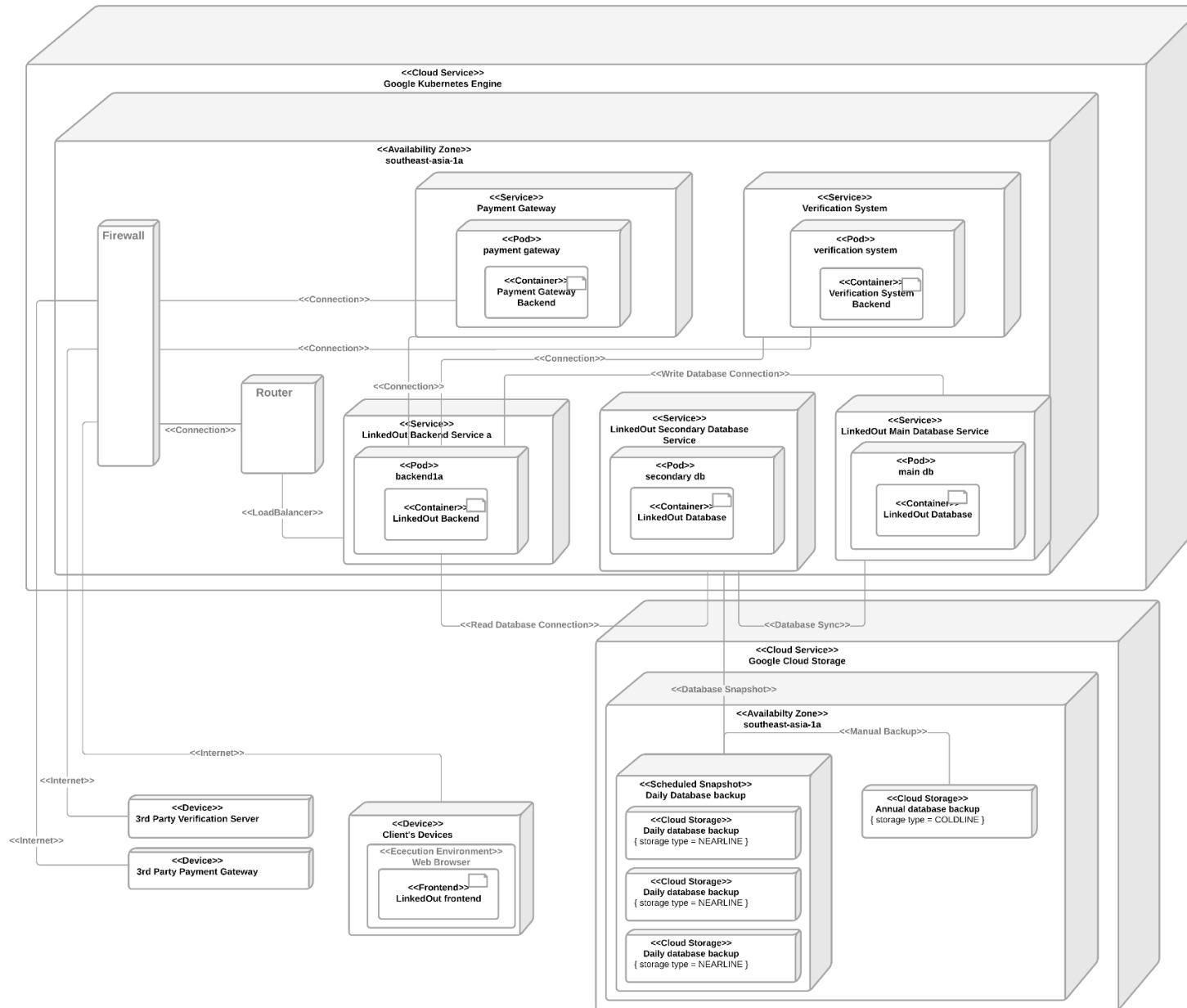


Figure 48 Low-cost Deployment Diagram

23) Hardware and software specification

จาก Deployment Diagram จะเห็นได้ว่ามีการแบ่ง Architecture ออกเป็น 3 ชั้นขึ้นไปโดย Deploy บน Google Kubernetes Engine และบนเครื่องคอมพิวเตอร์ส่วนตัวหรือสมาร์ตโฟนของผู้ใช้งาน เพื่อให้การใช้งานเป็นไปได้สะดวกและถูกต้องตาม requirement เราจึงได้เขียน specification ทั้ง software และ hardware ที่ความมีเป็นอย่างน้อยเพื่อให้ระบบทำงานได้ครบถ้วนทุกฟังก์ชัน โดยดูจากค่าใช้บริการ cloud service และประสิทธิภาพในการใช้งานเป็นหลัก

Specification	Client	Backend Server	Database Server
Stack of Development	<ul style="list-style-type: none"> - React.js with Babel and Webpack (ES6 targeted) - Axios 	<ul style="list-style-type: none"> - Node.js 14-lts with Koa framework - nginx 	<ul style="list-style-type: none"> - MySQL 8.0 - for monitoring, use MySQL Workbench remotely
Execution Environment	<ul style="list-style-type: none"> - Web browser มาตรฐานที่สามารถรัน HTML5, CSS3, ES6 เป็นอย่างน้อย เช่น Browser ในกลุ่ม Chromium (Google Chrome, Microsoft Edge, etc.) , Browser ในกลุ่ม Firefox (Mozilla Firefox, etc.) , Safari, etc. 	<ul style="list-style-type: none"> - Container ในระบบ Docker ชิ้นรันบน x64 	<ul style="list-style-type: none"> - Container ในระบบ Docker ชิ้นสามารถ bind volume เพื่อนำข้อมูลใน Database ออกจาก container ได้
Platform / Operating System	<ul style="list-style-type: none"> - ไม่จำกัด Operating System ขอเพียงสามารถรัน Web Browser ได้ตาม minimum requirement 	<ul style="list-style-type: none"> - Google Kubernetes Engine โดย Deploy ผ่าน service - มีการใช้ Horizontal Pod Autoscaler โดยใช้ CPU Usage เป็น metrics 	<ul style="list-style-type: none"> - Google Kubernetes Engine โดย Deploy ผ่าน service - มีการใช้ Horizontal Pod Autoscaler (กรณี High-cost implementation) โดยใช้ CPU Usage เป็น metrics - bind volume เข้าสู่ pod โดยใช้ Persistent Volume Claim (PVC) เพื่อเชื่อมกับ persistent disk ของ GCE
Hardware / Resource Allocation	<ul style="list-style-type: none"> - 2GB free memory at usage time - 250MiB free storage 	<ul style="list-style-type: none"> - 2 Core CPU/pod Allocation - 4GB Memory/pod Allocation 	<ul style="list-style-type: none"> - 2 Core CPU/pod Allocation - 4GB Memory/pod Allocation

	- 2.0GHz CPU with at least 2 cores		
Network Requirement	<ul style="list-style-type: none"> - เปิดพอร์ต 443 (HTTPS) - สามารถเชื่อมต่ออินเทอร์เน็ต โดยมีความเร็วการเชื่อมต่อ 384 kbps เป็นอย่างต่ำ 	<ul style="list-style-type: none"> - เปิดพอร์ต 443 สำหรับเชื่อมต่อกับอินเทอร์เน็ต 	<ul style="list-style-type: none"> - เปิดพอร์ต 3306 สำหรับเชื่อมต่อภายใน

Table 10 Hardware and software specification

24) Verifying and validating the physical architecture layer

ทั้ง Deployment diagram และ Network Diagram นั้นตรงตามและรองรับทุก functional requirement และ non-functional requirement ยกตัวอย่างเช่น การทำ Snapshot database เพื่อทำตาม requirement ที่ว่าต้องสำรองข้อมูลทั้งที่สำคัญและไม่สำคัญที่ระยะเวลาต่าง ๆ และการรวม traffic ไปที่ router เพื่อทำ firewall และ set up TLS ตาม non-functional requirement เป็นต้น

ทุกๆโครงสร้างที่มีอยู่ใน deployment diagram สามารถ map กับใน network diagram ในทางกลับกัน ทุกๆโครงสร้างที่มีอยู่ใน network diagram ต่างสามารถอธิบายได้ด้วยโครงสร้างใน deployment diagram เช่นกัน โดยมี component ทั้ง 3 คือ Frontend, Backend, Database และตัว Snapshot ครอบคลุมทุก component

กำหนดของ Hardware Specification กับ Software Specification เป็นไปตาม non-functional requirement ทุกประการ เช่น เรื่องของ Hardware Specification ขั้นต่ำของ Client เพื่อให้สามารถใช้งานตาม functional requirement และ non-functional requirement ได้ครบถ้วน รวมถึงได้ตรวจสอบความเหมาะสมของ Hardware Specification กับ Software Specification / Stack of Development แล้ว เช่น Node.js 14-lts กับ 2 core CPU, 4GB Memory ซึ่งไม่มากไม่น้อยเกินไป เป็นต้น

25) Design Impact

1. ผลกระทบด้านสิ่งแวดล้อม (Environment)

- ระบบใช้การเก็บข้อมูลแบบ Cloud ทำให้ลดประมาณขยายที่เกิดจากการซื้อที่เก็บข้อมูลแบบอื่นๆได้

2. ผลกระทบด้านเศรษฐกิจ (Economic)

- ระบบช่วยให้ขั้นตอนการจัดทำงานและการจ้างงานเป็นไปได้อย่างรวดเร็ว ส่งเสริมการจ้างงาน ช่วยให้เกิดเม็ดเงินหมุนเวียนในระบบเศรษฐกิจ

3. ผลกระทบโดยรวม (Global)

- ระบบสนับสนุนการใช้ smart device และ เทคโนโลยีต่าง ๆ เพื่อสนับสนุนให้วงการเทคโนโลยีของโลกก้าวหน้ามากยิ่งขึ้น

4. ผลกระทบต่อสังคม (Social)

- การใช้เทคโนโลยีต่างๆในการให้บริการต่อลูกค้าทำให้งานของพนักงานลดลง ทำให้พนักงานสามารถใช้เวลา กับครอบครัวได้มากยิ่งขึ้น

26) Conclusion

ในเอกสารฉบับนี้ เราได้ตรวจสอบ System Design Constraints เพื่อตรวจสอบว่าโครงการของเรามีข้อจำกัดหรือมีขอบเขตมากแค่ไหนเพื่อประกอบการตัดสินใจต่าง ๆ ในการออกแบบระบบ เพื่อที่จะได้มีออกแบบส่วนได้ส่วนหันของระบบมากเกินไปหรือน้อยเกินไปจากที่ควรจะเป็น ช่วยในเรื่องของการ optimize ทรัพยากรต่าง ๆ ที่ต้องใช้

จากนั้นเราได้ทำการ brainstorm โดยทำการทบทวนทุกทุกรายละเอียดในทุกทุก class และ use case เพื่อออกแบบระบบโดยรวมให้เป็นโครงสร้างที่ง่ายต่อการจัดการและการ scale แล้วจึงเพิ่มรายละเอียดในส่วนของ specification, partition, normalization, optimization, coupling, cohesion เพื่อเพิ่มความ consistent ในด้าน cohesion และ coupling รวมถึง optimize การส่ง message ระหว่าง class ให้มีความเป็น modular มากที่สุด

จากนั้นเราได้ทำการออกแบบ Human-Computer Interaction โดยเริ่มจากการเขียน use case description ประเภท detail, real และทำ use scenario เพื่อทำความเข้าใจ path ที่ user ต้องใช้ใน application ของเราโดยใช้ use case description ประเภท detail, essential และจึงเขียน windows navigation diagram เพื่อบอกว่าในแต่ละหน้าของ application ของเรา มีจุดเชื่อมต่ออย่างไร

หลังจากทำความเข้าใจในระบบของเราแล้ว เราได้เริ่มออกแบบตัวระบบจริง ๆ ซึ่งประกอบไปด้วย user interface โดยยึดตาม user interface design standard และ user interface design principle จากนั้นจึงสร้าง user interface prototype และเราได้ทำการตรวจสอบทั้ง functional requirement, non-functional requirement เพื่อออกแบบระบบของ我们在ใช้ physical ให้ตรงตาม requirement ทั้งหมด

System installation and operation

1) Migration plan

1.1) Conversion Plan

1.1.1) Hardware Purchase and Installation

เนื่องจาก Architecture ของเราใช้ระบบภายใน Google Cloud Platform เกือบทั้งหมด จึงไม่มีความจำเป็นต้องซื้อ Hardware สำหรับ Architecture โดยจะจัดให้มี billing account เพื่อจ่ายเงินค่าใช้บริการ Architecture บน Google Cloud Platform

สำหรับการ Allocate Resource บน Google Kubernetes Engine ในเบื้องต้นจะจำกัดให้มี availability zone ละ 2 instances โดยใช้เป็น n2-standard-2 โดยสามารถขยายเพิ่มในภายหลังได้ตามการเติบโตของผู้ใช้งาน ในด้านของ Client ผู้ใช้งานจำเป็นต้องมี PC หรือ smart phone คู่กับ web browser ที่รองรับ HTML5, CSS3, ES6 เป็นของส่วนบุคคลจะเข้า Application ได้ จึงไม่มีความจำเป็นต้องซื้อหรือ install ใด ๆ

1.1.2) Software Installation

สำหรับการ Deploy Application ภายใน GCP นั้นไม่มีความจำเป็นต้องลง software สำหรับการ deploy ได้ ๆ เป็นพิเศษ แต่สำหรับเครื่องคอมพิวเตอร์ Development ที่ผู้พัฒนาจำเป็นต้องใช้ กำหนดให้ลงโปรแกรม gcloud ในเครื่องของผู้พัฒนาเพื่อให้การพัฒนา Application มีความสะดวกมากขึ้น

1.1.3) Data Conversion

เนื่องจาก Application ของเราสร้างขึ้นมาใหม่ ไม่ได้ต่อยอดมาจากระบบเดิมที่มีอยู่แล้ว จึงไม่ต้องทำ Data Conversion

1.2) Conversion Dimension

1.2.1) Conversion Strategy

Parallel, Phased, Modular

1.2.2) Conversion Style

ทีมผู้พัฒนาได้เลือกรูปแบบการเปลี่ยนแปลงแบบ Parallel ในการเปลี่ยนแปลงระบบ เนื่องจากระบบ Linked Out ที่สร้างขึ้นมาใหม่ไม่ได้พัฒนาจากระบบที่มีอยู่ ในระยะของการเปลี่ยนแปลงจึงสามารถใช้งานคู่ขนานไปกับการรับสมัครแบบเดิม หรือซ่องทางอื่น ๆ ที่มีอยู่แล้วได้

1.2.3) Conversion Location

ทีมผู้พัฒนาเลือกการเปลี่ยนแปลงระบบแบบ Phased เพื่อให้สอดคล้องกับ SDLC Process ของทีมผู้พัฒนา โดยเลือกกลุ่มบริษัทที่รับสมัครงานนำร่องบางกลุ่มก่อน เพื่อสำรวจความต้องการและข้อคิดเห็นของผู้ใช้ระบบเพิ่มเติม ให้ระบบเกิดการพัฒนาต่ออยู่ต่อไป และเพื่อป้องกันความเสียหายที่เกิดขึ้นจากข้อผิดพลาดที่อาจมีอยู่ในระบบ จากนั้น จึงขยายไปยังกลุ่มผู้จัดทำงานอื่น ๆ ต่อไป

1.2.4) Conversion Modules

ทีมผู้พัฒนาเลือกใช้การติดตั้งระบบแบบ Modular เพื่อให้สอดคล้องกับ SDLC ของทีมผู้พัฒนา แบบ Phased Development ทำให้สามารถติดตั้งระบบทีละส่วนได้ นอกเหนือไปความสามารถในการติดตั้งของผู้ใช้งานต่อระบบในส่วนที่ได้รับการติดตั้งแล้วมา ปรับปรุงส่วนที่ยังไม่ได้รับการติดตั้ง โดยมีแผนงานจะติดตั้งระบบดังนี้

- 1) ส่วนการค้นหาประกาศงาน – ระบบรับสมัครงาน
- 2) ส่วนการโฆษณาภายในระบบ

2) post-implementation activities (system support and maintenance)

หลังจากการออกแบบระบบเสร็จสิ้น และแอปพลิเคชันถูกนำไปใช้งานจริง โครงการ Linked Out ได้มีการเตรียมการ Post-implementation activities เพื่อเป็นบริการหลังการติดตั้งแอปพลิเคชัน ประกอบไปด้วย System support, System Maintenance และ Project assessment

2.1 System Support

2.1.1 Online support

ระบบจะมีบริการสำหรับช่วยเหลือผู้ใช้งานแอปพลิเคชันแบบออนไลน์ ผ่านทางเวปไซต์ ประกอบไปด้วย คำถามที่พบบ่อย (Frequently Asked Question หรือ FAQ) เพื่อให้ผู้ใช้งานสามารถแก้ปัญหาต่าง ๆ ในเบื้องต้นได้ หากปัญหาที่ผู้ใช้งานพบมีความซับซ้อนกว่าปัญหาที่พบบ่อยใน FAQ ผู้ใช้งานแอปพลิเคชันสามารถติดต่อกับทีมงานของโครงการผ่านทาง E-mail โดยจะมีทีมงานที่มีหน้าที่คอยช่วยเหลือผู้ใช้งานโดยตรง

2.1.2 Admin support

หากผู้ใช้งานมีปัญหาในขั้นตอนการยืนยันตัวตน ผู้ใช้งานสามารถติดต่อ Admin ได้โดยตรง ไม่ว่าจะเป็นสมาชิกที่ผู้หางาน (Member) หรือ บริษัทที่หางาน (Recruiter) โดยการติดต่อผ่านทาง E-mail หรือ รับการแจ้งข้อผิดพลาดของระบบต่าง ๆ ที่อาจเกิดขึ้น Admin จะนำปัญหาที่เกิดขึ้น มาปรึกษากับทางทีมผู้พัฒนาเพื่อคุ้มครองระบบของโครงการต่อไป

2.2 System Maintenance

2.2.1 Users Feedback

หากผู้ใช้งานพบข้อผิดพลาดของระบบ หรือ ข้อแนะนำในการพัฒนาระบบ ผู้ใช้งานสามารถแจ้งมาทาง Admin เพื่อให้แก้ไขข้อบกพร่องต่อไปได้ โดย Admin จะเตรียมการกับทีมผู้ดูแลระบบ

2.2.2 Closed for maintenance

ทางทีมผู้ดูแลระบบจะมีการปิดปรับปรุงระบบเพื่อแก้ไขปัญหาต่าง ๆ ในระบบ และ พัฒนา feature ต่าง ๆ ของระบบ ตั้งแต่เวลา 00.00 น. – 01.00 น. เป็นระยะเวลา 1 ชั่วโมง ทุก ๆ 2 อาทิตย์ ในวันอาทิตย์

2.3 Project Assessment

การประเมินโครงการถูกจัดขึ้นเพื่อวิเคราะห์ข้อดีของระบบ ข้อบกพร่องต่าง ๆ ของระบบ และเพื่อหาแนวทางในการพัฒนาระบบท่อไป ทีมประเมินจะประเมินระบบว่าควรพัฒนา feature อะไรต่อไป ซึ่งการประเมินจะมีการจัดทุก ๆ เดือน โดยมีการประเมินทั้งหมด 2 ส่วน ได้แก่

2.3.1 Project team review

การประเมินทีมงานของโครงการมีขึ้นเพื่อพัฒนาระบบและจัดการกับข้อผิดพลาดต่าง ๆ ของระบบที่อาจจะมีการเกิดขึ้นได้ในอนาคต

ทุก ๆ คนในทีมมีความสามารถด้านต่าง ๆ กัน การประเมินร่วมกันทำให้สามารถมองเห็นปัญหาต่าง ๆ ได้รอบด้าน และ ทำให้โครงการสามารถพัฒนาต่อไปได้อย่างเข้มแข็ง

เมื่อมีการประชุมร่วมกันทำให้สามารถลดปัญหาที่อาจเกิดจากการสื่อสารที่ผิดพลาดได้ เพราะสามารถถามข้อสงสัยต่าง ๆ ได้เลยในที่ประชุม

การประชุมร่วมกันทำให้ทีมงานทุกคนมองเห็นภาพรวมของระบบไปในทิศทางเดียวกัน ทำให้ระบบสามารถพัฒนาไปได้ในทิศทางที่ทีมต้องการ

สามารถคำนวณเวลาที่ใช้ในการพัฒนาระบบในขั้นตอนต่าง ๆ เช่น การเพิ่ม feature ทำให้สามารถจัดการแบ่งงาน และ กำหนด dead line ของงาน ได้อย่างมีประสิทธิภาพ

2.3.2. System review

การประเมินระบบจัดขึ้นเพื่อให้สามารถเบรียบเทียบระบบที่ตั้งเป้าหมายไว้ กับระบบจริง ๆ เพื่อวิเคราะห์หาข้อดี ข้อเสียของระบบ

List of teammate contribution

Name	ID	Contribution
ชาญชัย รัตนะศิรากุล	6130115221	Class Diagram, Use case diagram, Component diagram, Document format
ธัมม์พิรัตถ์ ติระนาทวิทยากร	6130247421	Class Diagram, Use case diagram, Component diagram, CRUDE matrix, Conversion, Contract Specification, Method Specification, CRC Cards, Conversion Dimension, Design Impact, Document format
ธีรวัฒน์ แก่งศิลาลัย	6130257721	Class Diagram, Use case diagram, Component diagram, Contract Specification, Method Specification, CRC Cards, Design Impact
ปุณย์วัชร์ รุจิพิรานันท์	6131027921	Class Diagram, Use case diagram, Physical Architecture Design and Specification, Requirements, Conversion Plan, Network Diagram, Deployment Diagram
พศรัตน์ แตงไหญ	6131029121	Class Diagram, Use case diagram, Component diagram, CRC Cards
พศรัตน์ ไม้เหลือง	6131030721	Class Diagram, Use case diagram, Component diagram, User Interface Design, WND, Conversion Dimension
ภูมิพัฒน์ จิรจัล	6131037121	Class Diagram, Use case diagram, Component diagram, Contract Specification, CRC Cards, Conversion Dimension
สรัชญ์ กฤตวีรนันท์	6131044521	Class Diagram, Use case diagram, Component diagram, Validate Class Diagram, Document format

Table 11 List of teammate contribution