

**ບົດໂຄງການຈົບຊັ້ນລະດັບປະລິນຍາຕີວິທະຍາສາດ**

**ສາຂາ ວິທະຍາສາດຄອມພິວເຕີ**

**ລະບົບຈອງປີ້ລົດເມສາຍໃຕ້ອອນລາຍ**

**Southern Bus Ticket Online Booking System**

ທ້າວ ມະໂນພອນ ມະໂນກຸນ

ທ້າວ ພອນຄຳ ແກ້ວມະນີ

ຄະນະວິທະຍາສາດທຳມະຊາດ

ສົກສຶກສາ 2020-2021

**ບົດໂຄງການຈົບຊັ້ນລະດັບປະລິນຍາຕີວິທະຍາສາດ**

**ສາຂາ ວິທະຍາສາດຄອມພິວເຕີ**

**ລະບົບຈອງປີ້ລົດເມສາຍໃຕ້ອອນລາຍ**

**Southern Bus Ticket Online Booking System**

**ນັກສຶກສາ:**

**ທ້າວ ມະໂນພອນ ມະໂນກຸນ**

**ທ້າວ ພອນຄຳ ແກ້ວມະນີ**

**ອາຈານຜູ້ນຳພາ:**

**ຄະນະວິທະຍາສາດຄອມພິວເຕີ**

**ພາກວິຊາ ວິທະຍາສາດທຳມະຊາດ**

**ສົກສຶກສາ 2020-2021**

**Final Project of Bachelor Degree of Natural Sciences in Computer science**

**ລະບົບຈອງປີ້ລົດເມສາຍໃຕ້ອອນລາຍ**

**Southern Bus Ticket Online Booking System**

**Students:**

**Mr. Manophone MANOKOUN**

**Mr Phonekham KEOMANY**

**Advisor:**

**Faculty of Natural Sciences**

**Department of Computer Sciences**

**Academic Year 2020-2021**

**ຄໍານໍາ**

ໃນປັດຈຸບັນ ເທັກໂນໂລຊີນັບມື້ນັບທັນສະໄໝຍິ່ງຂຶ້ນ ແລະ ມີຄວາມກ້າວໜ້າຢ່າງບໍ່ຢຸດຢັ້ງເຊິ່ງມັນມີບົດບາດສໍາຄັນໃນການດໍາລົງຊີວິດຂອງພວກເຮົາ, ໃນນີີ້ການນໍາໃຊ້ເທັກໂນໂລຊີເຂົ້າມາຊ່ວຍໃນການຈັດການຂໍ້ມູນໂດຍສະເພາະແມ່ນການຈັດການກ່ຽວກັບຂໍ້ມູນຖືວ່າແມ່ນວຽກງານໜຶ່ງທີ່ສໍາຄັນຕໍ່ກັບບັນດາວິສາຫະກິດ, ທຸລະກິດ ແລະ ອົງກອນຕ່າງໆ ເພື່ອເຮັດໃຫ້ການຈັດການຂໍ້ມູນຕ່າງໆມີຄວາມວ່ອງໄວທັນຕໍ່ສະພາບການ ແລະ ສາມາດເກັບຮັກສາຂໍ້ມູນຕ່າງໆທີ່ມີຄວາມສໍາຄັນຕໍ່ ກັບອົງກອນໃຫ້ມີຄວາມປອດໄພຂຶ້ນ.

ຈຸດປະສົງຫຼັກຂອງການສ້າງລະບົບລະບົບຂາຍປີ້ລົດເມອອນໄລຂອງສະຖານຂົນສົ່ງ ໂດຍສານສາຍໃຕ້ຂຶ້ນມາແມ່ນເພື່ອຈະນໍາເອົາເທັກໂນໂລຊີທີ່ທັນສະໄໝເຂົ້າມາຊ່ວຍໃນການຂາຍປີ້ແລະ ຈັດການຂໍ້ມູນ ໃຫ້ມີຄວາມສະດວກ, ເພີ່ມປະສິດທິພາບໃນການເຮັດວຽກໃຫ້ມີຄວາມຖືກຕ້ອງ ແລະ ຊັດເຈນທີ່ສຸດ.

ເຖິງຢ່າງໃດກໍ່ຕາມ ພວກຂ້າພະເຈົ້າຫັວງຢ່າງຍິ່ງວ່າບົດໂຄງການຈົບຊັ້ນນີ້ຈະມີຄຸນປະໂຫຍດບໍ່ຫຼາຍກໍ່ໜ້ອຍຕໍ່ນັກສຶກສາລຸ້ນຕໍ່ໄປ ສໍາຫຼັບຜູ້ທີ່ສົນໃຈສ້າງ ຫຼື ພັດທະນາລະບົບທີ່ກ່ຽວ ຂ້ອງ. ພ້ອມດຽວກັນກໍ່ຫັວງວ່າລະບົບດັ່ງກ່າວນີ້ ຈະຊ່ວຍອໍານວຍຄວາມສະດວກໃນການບໍລິການ ແລະ ຈັດການຂໍ້ມູນຕ່າງໆຂອງທາງສະຖານີຂົນສົ່ງໂດຍສານສາຍໃຕ້. ໃນລະບົບທີ່ພວກຂ້າພະເຈົ້າໄດ້ສ້າງຂຶ້ນມານີ້ບໍ່ອາດຫຼີກລ້ຽງຂໍ້ຜິດພາດໄດ້, ສະນັ້ນ ພວກຂ້າພະເຈົ້າຈຶ່ງຂໍອະໄພມາຍັງບັນດາທ່ານ ແລະ ຍອມຮັບຄໍາຕໍານິຕິຊົມຈາກທ່ານ ແລະ ນໍາມາແກ້ໄຂ ເຮັດໃຫ້ປຶ້ມຫົວນີ້ສົມ ບູນຂຶ້ນ



ສາທາລະນະລັດປະຊາທິປະໄຕປະຊາຊົນລາວ

ສັນຕິພາບ ເອກະລາດ ປະຊາທິປະໄຕ ເອກະພາບ ວັດທະນະຖາວອນ

ມະຫາວິທະຍາໄລແຫ່ງຊາດ

ຄະນະວິສະວະກຳສາດ ເລກທີ..................../.............

ນະຄອນຫຼວງວຽງຈັນ,ວັນທີ...............................

**ໃບຮັບຮອງບົດໂຄງການຈົບຊັ້ນ**

**ລະບົບຈອງປີ້ລົດເມສາຍໃຕ້ອອນລາຍ**

**Southern Bus Ticket Online Booking System**

**ນັກສຶກສາ:**

ທ້າວ ມະໂນພອນ ມະໂນກຸນ ລະຫັດນັກສຶກສາ 204N0002.19

ທ້າວ ພອນຄຳ ແກ້ວມະນີ ລະຫັດນັກສຶກສາ 204N0025.19

**ອາຈານຜູ່ນຳພາ:**

ອຈ. …………………………………

ສະພາວິຊາການຂອງຄະນະວິທະຍາສາດທຳມະຊາດ ອະນຸມັດໃຫ້ບົດໂຄງການຈົບຊັ້ນສະບັບນີ້ເປັນສ່ວນໜື່ງຂອງການສຶກສາຕາມຫຼັກສູດປະລິນຍາຕີ (ຕໍ່ເນື່ອງ) ວິທະຍາສາດທຳມະຊາດ ສາຂາ ວິທະຍາສາດຄອມພິວເຕີ

**ຄະນະກຳມະການປ້ອງກັນບົດໂຄງການຈົບຊັ້ນ**

ຄັ້ງວັນທີ ເດືອນ ປິ 2021

1. ອຈ. .....…..........………………….ປະທານກຳມະການ
2. ອຈ. .............…………….......ກໍາມະການ
3. ອຈ. ......................……………........…...ກຳມະການ

**ຄະນະບໍດີ ຫົວໜ້າພາກວິຊາ ວິທະຍາສາດຄອມພິວເຕີ**

# **ບົດຄັດຫຍໍ້**

ບົດໂຄງການຈົບຊັ້ນນີ້ເປັນການສຶກສາ ພັດທະນາລະບົບການຈອງປີ້ລົດເມແບບອອນໄລເຊັ່ນ: ການຈອງປີ້ລົດແບບອອນລາຍ,ການຈັດການຂໍ້ມູນ, ລາຍງານຂໍ້ມູນຕ່າງໆ, ລວມທັງການຂາຍປີ້ແມ່ນຍັງໃຊ້ແບບຈົດ ແລະ ຜູ້ໂດຍສານທີ່ຕ້ອງການຈອງປີ້ລວງໜ້າ ຕ້ອງໄດ້ໂທຫາພະນັກງານຂາຍປີ້ເພື່ອຈອງ ເຊິ່ງເຮັດໃຫ້ການບໍລິການມີການຊັກຊ້າ ແລະ ຂໍ້ມູນຍັງມີການຕົກເຫ່ຍເສຍຫາຍ ດັ່ງນັ້ນ , ພວກຂ້າພະເຈົ້າຈຶ່ງເຫັນຄວາມສໍາຄັນຂອງບັນຫາ ຈຶ່ງມີແນວຄວາມຄິດທີ່ຈະສ້າງລະບົບຈອງປີລົດເມ ແບບອອນໄລ ຂອງສະຖານນີຂົນສົ່ງໂດຍສານສາຍໃຕ້ນັ້ນຂຶ້ນມາ ເພື່ອຊ່ວຍຫຸດຜ່ອນຄວາມຫຍຸ້ງຍາກໃນການຈອງປີ້ລົດ, ຈັດເກັບຂໍ້ມູນ, ຫຸດຜ່ອນຄວາມຊັກຊ້າໃນການຈັດການຂໍ້ມູນ, ເຮັດໃຫຂໍ້ມູນມີຄວາມເປັນລະບຽບຮຽບຮ້ອຍ ແລະ ເພື່ອໃຫ້ມີຄວາມສະດວກວ່ອງໄວຕໍ່ການຄົົນຫາຂໍ້ມູນ. ດັ່ງນັ້ນ, ຈື່ງຮຽກຮ້ອງໃຫ້ນໍາເອົາລະບົບການຈັດການຖານຂໍ້ມູນທີ່ມີປະສິດທິພາບສູງເຂົາມາຊ່ວຍໃນການບໍລິຫານເກັບກຳເອົາບັນດາຂໍ້ມູນຕ່າງໆ ທີ່ເພີ່ມຂື້ນນັ້ນໃຫ້ມີຄວາມເປັນລະບຽບຮຽບຮ້ອຍ ຄົບຖ້ວນ ແລະ ສົມບູນກ່ອນເກົ່າ ເຊີ່ງນອກຈາກນຳເອົາວິຊາການ ວິເຄາະ ແລະ ອອກແບບລະບົບມານຳໃຊ້ແລ້ວກໍ່ໄດ້ນຳໃຊ້ໂປແກມ Microsoft Visio 2016 ໃຊ້ແຕ້ມແຜນວາດການໄຫຼຂໍ້ມູນ (DFD, ER, Flowchart) , Adobe XD ໃຊ້ອອກແບບ UX/UI , Studio 3T For MongoDB ແລະ Moon Modeler ໃຊ້ຈັດການຖານຂໍ້ມູນ ແລະ ອອກແບບ Database Model ,Visual Studio Code ໃຊ້ຂຽນໂຄດດ້ວຍພາສາ JavaScript (ReactJS, NodeJS, GraphQL)

**Abstract**

Southern Station Bus Booking Online System was developed to make the sale system be modern, serve customers more efficiently and quickly. The development of the Southern Station Bus Booking Online System consists of 2 main parts which are software and hardware. The software part we use phpMyAdmin for storing data for the program. For the database, we use Mysql Sublime text 3 for designing forms and writing the program to control and connect to the database. For the reports, we use Crystal Reports for building reports, We use Microsoft Visio 2016 for drawing data flow diagrams, flowchart and entity relationship diagrams (ER-Diagram). The hardware part we use 2 laptops for developing the application. The results from Southern Station Bus Booking Online System are a system that will reduce redundancy and complexity of the data, to facilitate the customers. Making the users of the system be more convenient and have accurate information. Also, the users can order, serve the customers, search information and get reports quickly and accurately with accurate information.

# **ສາລະບານ**

**ໜ້າທີ**

ໜ້າປົກໃນ

ໃບຮັບຮອງບົດໂຄງການຈົບຊັ້ນ

ບົດຄັດຫຍໍ້ພາສາລາວ**........................................................................................................**i

ບົດຄັດຫຍໍ້ພາສາອັງກິດ**....................................................................................................**ii

ຄຳສະແດງຄວາມຂອບໃຈ**................................................................................................**iii

ສາລະບານ**………………………………………………………………………………….…**iv

ສາລະບານຕາຕະລາງ **..**x

ສາລະບານຮູບ**...............................................................................................................**xi

[ພາກທີ 1](#_Toc47627985) ບົດນຳສະເໝີ**……….……………………………………………………….……..**1

1.1 [ປະຫວັດຄວາມເປັນມາ…..…..……………………………………………...……1](#_Toc47627987)

[1.2 ຈຸດປະສົງຂອງການສືກສາ……………………………………………….……….1](#_Toc47627988)

[1.3 ຂອບເຂດຂອງການສຶກສາ………………………………………………….…….2](#_Toc47627989)

[1.4 ປະໂຫຍດທີ່ຄາດວ່າຈະໄດ້ຮັບ………………………….…………………….……2](#_Toc47627990)

[1.4.1 ປະໂຫຍດຕໍ່ນັກສຶກສາ**………………………………………….……………….**2](#_Toc47627991)

[1.4.2 ປະໂຫຍດຕໍ່ອົງກອນ**……………………………………………………………**2](#_Toc47627992)

[1.5 ແຜນການດໍາເນີນງານ 4](#_Toc47627993)

[ບົດທີ 2](#_Toc47627994) [ທິດສະດີພຶ້ນຖານທີ່ກ່ຽວຂ້ອງ**……………………………………………………………**5](#_Toc47627995)

[2.1.1 ທິດສະດີທີ່ກ່ຽວຂ້ອງ 5](#_Toc47627996)

[2.1.2 ຄວາມໝາຍຂອງລະບົບ**…………………….…………………………………….**5](#_Toc47627998)

[2.1.3 ບົດບາດ ແລະ ຄວາມສໍາຄັນຂອງລະບົບ….**……………..………………..……….**5](#_Toc47627999)

**ສາລະບານຕາຕະລາງ**

**ໜ້າທີ**

ຕາຕະລາງທີ 1.1 ໄລຍະເວລາໃນການປະຕິບັດງານ 4

ຕາຕະລາງທີ 3.1 ບັນຫາ ແລະ ສາເຫດ 26ຕາຕະລາງທີ 3.2 ຕາຕະລາງຈັດການຂໍ້ມູນຜູ້ໃຊ້…………………………...…………………...39

ຕາຕະລາງທີ 3.3 ຕາຕະລາງຫ້ອງພັກ 39

ຕາຕະລາງທີ 3.4 ຕາຕະລາງຈັດການຂໍ້ມູນປະເພດຫ້ອງພັກ 40

ຕາຕະລາງທີ 3.5 ຕາຕະລາງຈັດການຂໍ້ມູນລູກຄ້າ 40

ຕາຕະລາງທີ 3.6 ຕາຕະລາງCheck in 41

ຕາຕະລາງທີ 3.7 ຕາຕະລາງຂໍ້ມູນລາຍລະອຽດຂອງການເຂົ້າພັກ 41

ຕາຕະລາງທີ 3.8 ຕາຕະລາງBooking 42

ຕາຕະລາງທີ 3.9 ຕາຕະລາງລາຍລະອຽດການຈອງ 42

**ສາລະບານຮູບ**

**ໜ້າທີ**

ຮູບທີ 1: ນັກວິເຄາະລະບົບ (System Analysis) ..9

ຮູບທີ 2: ຮູບຕາຕະລາງສັນຍາລັກ Data Flow Diagram .11

ຮູບທີ 3:ຮູບ Strong Entity .13

ຮູບທີ 4: ຮູບ Weak Entity **…………………………………………………………………..**13

ຮູບທີ 5: ຮູບ Attribute **.**14

ຮູບທີ 6: ຮູບ Attribute.14

ຮູບທີ 7: ຮູບ Attribute.15

ຮູບທີ 8: ຄວາມສໍາພັນແບບໄບນາຣີ (Binary Relationships) **………………………………..**15

ຮູບທີ 9: ຄວາມສໍາພັນແບບເທີ້ນາຣີ (Ternary Relationships) **……………………………...**16

ຮູບທີ 10: ຮູບການປຽບທຽບແຜນວາດການໄຫຼຂໍ້ມູນທີ່ບໍ່ຖືກຕ້ອງ ແລະ ຖືກຕ້ອງ.17

ຮູບທີ 11: ຮູບVersion 289 SQL Server**…………………** .22

ຮູບທີ 3.1 ສະແດງຂະບວນການດໍາເນີນວຽກລະບົບປັດຈຸບັນ**………. .**25

ຮູບທີ 3.2 ສະແດງຂະບວນການດໍາເນີນວຽກລະບົບໃໝ່.28

**ສາລະບານຄຳຫຍໍ້**

ຄວທ = ຄະນະວິທະຍາສາດທຳມະຊາດ

ມຊ = ມະຫາວິທະຍາໄລແຫ່ງຊາດ

DFD = Data Flow Diagram

E-R Model = Entity Relationship Model

E-R Diagram = Entity Relationship Diagram

PHC = Process Hierarchy Chart

SDLC = System Development Life Cycle

SA = System Analyst

SQL = Structured Query Language

VB.net = Visual Basic Studio

CASE = Computer Aided Software Engineering

DBMS = Database Management System

DML = Data Manipulation Language

DCL = Data Control Language

**ບົດທີ 1**

# **ບົດສະເໜີ**

## ຄວາມສຳຄັນຂອງບັນຫາ.

ອີງໃສ່ເພື່ອຄວາມສະດວກສະບາຍຂອງປະຊາຊົນລາວເຮົາ ແລະ ຄົນຕ່າງປະເທດ, ນັກທ່ອງທ່ຽວ, ພະນັກງານ, ພໍ່ຄ້າຊາວຂາຍຕະຫຼອດຮອດພໍ່ແມ່ປະຊາຊົນທີ່ເດີນທາງເຂົ້າມາທ່ອງທ່ຽວ ຫຼື ມາເຮັດວຽກເຮັດງານທີ່ນະຄອນຫຼວງວຽງຈັນຂອງພວກເຮົາເພື່ອໃຫ້ການໄປມາສະຖານທີ່ຕ່າງໆພວກເຮົາຈຶ່ງໄດ້ສ້າງຕັ້ງສະຖານີຂົນສົ່ງໂດຍສານນີ້ຂຶ້ນມາ.

ສະຖານີຂົນສົ່ງໂດຍສານສາຍໃຕ້ ເປັນບ່ອນບໍລິການຮັບ-ສົ່ງຜູ້ໂດຍສານ, ສິນຄ້າວັດຖຸສິ່ງ ຂອງ ແລະ ສັດ ຈາກຈຸດໜຶ່ງໄປຫາອີກຈຸດໜຶ່ງ ຊຶ່ງສະຖານນີຂົນສົ່ງໂດຍສານທາງໄກສາຍໃຕ້ນີ້ແມ່ນໄດ້ ສ້າງຕັ້ງຂຶ້ນໃນວັນທີ 1 ກັນຍາ 2016 ເຊິ່ງມີລາຍລະອຽດດັ່ງລຸ່ມນີ້: ທີ່ຕັ້ງ ແລະ ພາລະບົດບາດ ຂອງສະຖານນີຂົນສົ່ງທາງໄກສາຍໃຕ້ແມ່ນສະຖານນີໜຶ່ງຊຶ່ງຕັ້ງຢູ່ ບ້ານ ສະພັງມຶກ, ເມືອງ ໄຊທານີ, ນະຄອນຫຼວງວຽງຈັນ, ຖະໜົນເລກທີ 450 ປີ ໃກ້ກັບ ສີ່ແຍກໄຟແດງດົງໂດກ. ສະຖານນີຂົນສົ່ງທາງໄກສາຍໃຕ້ ປະກອບມີຫຼາຍໜ່ວຍງານຄື: ອຳນວຍການໃຫ່ຍມີ 1 ທ່ານ, ເລຂານຸການມີ 1 ທ່ານ, ໜ່ວຍງານແຜນການມີ 1 ທ່ານ, ໜ່ວຍງານຮັບ-ຈ່າຍເງິນມີ 1 ທ່ານ, ໜ່ວຍງານບໍລິການຂາຍປີ້ມີ 7 ທ່ານ, ໜ່ວຍງານຮັກສາຄວາມປອດໄພມີ 6 ທ່ານ,ໜ່ວຍງານບໍລິການເຮືອນພັກມີ 8 ທ່ານ ແລະ ບັນດາບໍລິສັດ ທິ່ເຂົ້າມາດໍາ ເນີນທຸລະກິດ ໃນສະຖານຂົນສົ່ງໂດຍສານປະກອບມີ: ບໍລິສັດ ຂົນສົ່ງໂດຍສານຈິດປະສົງ ຍອດນິຍົມ, ບໍລິສັດ ແສງສົມບູນ ຂົນສົ່ງໂດຍສານ, ບໍລິສັດ ແສນສະບາຍ ຂົນສົ່ງໂດຍສານ, ບໍລິສັດ ຈັນທະຈອນ ຂົນສົ່ງໂດຍສານ, ບໍລິສັດ ຈໍາປາສັກ ຂົນສົ່ງໂດຍສານ, ບໍລິສັດ ແສງຈະເລີນ ລົດຕຽງນອນ, ບໍລິສັດ ກຽງໄກ VIP, ບໍລິສັດ ສີທອນ ພວງປະເສີດ ລົດຕຽງນອນ. ນອກຈາກນີ້ ສະຖານຍີງ ມີສະຖານທີ່ພັກ , ຮ້ານຄ້າ, ຮ້ານຂາຍຍ່ອຍ, ຮ້ານອາຫານ ແລະ ສິ່ງອໍານວຍຄວາມສະດວກຕ່າງໆ ໄວ້ເພື່ອບໍລິການຜູ້ໂດຍສານທີ່ມາລໍຖ້າລົດໄປຈຸດໝາຍປາຍທາງ.​

## **ຈຸດປະສົງຂອງການສຶກສາ**

ວັດຖຸປະສົງຂອງບົດຫຼັກຂອງບົດຈົບຊັ້ນໃນຄັ້ງນີ້ແມ່ນອີງໃສ່ໃນການພັດທະນາ ແລະ ເພີ່ມປະສິດທິພາບໃນການບໍລິການຂອງສະຖານີຂົນສົ່ງໂດຍສານສາຍໃຕ້, ພວກຂ້າພະເຈົ້າຈຶ່ງໄດ້ມີແນວຄິດທຈະສ້າງ ແລະ ພັດທະນາ ລະບົບຈອງປີ້ລົດເມສາຍໃຕ້ອອນລາຍຂຶ້ນມາເພື່ອເເກ້ໄຂບັນຫາໃຫ້ກັບທາງສະຖານີຕາມວຽກຂອງເເຕ່ລະພະແນກເຖິງບັນຫາໃນການເຮັດວຽກດັ່ງນັ້ນ, ພວກຂ້າພະເຈົ້າຈຶ່ງໄດ້ກໍານົດເຫດຜົນຂອງການຄົ້ນຄ້ວາຕາມຈຸດປະສົງດັ່ງນີ້:

* ເພື່ອສຶກສາບັນຫາທີ່ເກີດຂຶ້ນໃນປະຈຸບັນ ແລະ ຄວາມຕ້ອງການຂອງລະບົບ.
* ເພື່ອສ້າງລະບົບຂາຍປີ້ລົດອອນໄລຂອງສະຖານນີຂົນສົ່ງໂດຍສານສາຍໃຕ້.
* ເພື່ອສ້າງຮູບແບບການຈັດການຂໍ້ມູນການໃຫບໍລິການ.
* ເພື່ອເຜີຍແຜ່ຂໍ້ມູນການຂາຍປີ້ລົດເມຂອງສະຖານຂົນສົ່ງໂດຍສານສາຍໃຕ້.
* ເພື່ອການລາຍງານໃຫ້ສະດວກ ແລະ ຖືກຕ້ອງ.

## **ຂອບເຂດການຄົ້ນຄວ້າ**

ລະບົບຈອງປີ້ລົດເມສາຍໃຕ້ແບບອອນລາຍ ເປັນລະບົບແບບ Client-Server Web Application ເຊິ່ງປະກອບດ້ວຍໜ້າວຽກຫຼັກດັ່ງນີ້:

* ຈັດການຂໍ້ມູນພື້ນຖານ (ຂໍ້ມູນພະນັກງານ, ຂໍ້ມູນລົດ, ຂໍ້ມູນປະເພດລົດ, ຂໍ້ມູນສາຍທາງ)
* ສະໝັກສະມາຊິກ
* ບໍລິການ (ຈອງປີ້, ອອກປີ້)
* ລາຍງານ : (ລາຍງານການຈອງ, ລາຍງານຂໍ້ມູນພະນັກງານ, ລາຍງານຂໍ້ມູນສາຍທາງ, ລາຍງານຂໍ້ມູນລົດ, ລາຍງານຂໍ້ມູນຊຳລະເງິນ)

## **ປະໂຫຍດທີ່ຄາດວ່າຈະໄດ້ຮັບ**

## ໃນການຂຽນບົດຈົບຊັ້ນໃນຄັ້ງນີ້ ຫຼັງຈາກສໍາເລັດໂຄງການນີ້ແລ້ວ Web Application ແມ່ນຕ້ອງໃຫ້ໄດ້ຮັບຜົນຕົວຈິງ ແລະ ສາມາດນໍາເຂົ້າມາໃຊ້ໃນວຽກງານຕົວຈິງໄດ້ຢ່າງແນ່ນອນ.

* ໄດ້ລະບົບຈອງປີ້ລົດເມແບບອອນລາຍຂອງສະຖານນີຂົນສົ່ງສາຍໃຕ້
* ໄດ້ລະບົບທີ່ຈະຊ່ວຍແກ້ໄຂບັນຫາການຈອງໄດ້ສະດວກ ແລະ ວ່ອງໄວຂຶ້ນ
* ໄດ້ລະບົບຊ່ວຍເພີ່ມຊ່ອງທາງໃນການຂາຍປີ້ໃຫ້ກັບຜູ້ປະກອບການ
* ມີລະບົບເຜີຍແຜ່
* ໄດ້ລະບົບທີ່ສາມາດສ້າງລາຍງານໄດ້ຢ່າງສະດວກ ແລະ ຖືກຕ້ອງ

# **ບົດທີ 2**

# **ທົບທວນເອກະສານ ແລະ ບົດຄົ້ນຄວ້າທີ່ກ່ຽວຂ້ອງ**

## ທົບທວນທິດສະດີທີ່ກ່ຽວຂ້ອງ

### ຄວາມຮູ້ພື້ນຖານກ່ຽວກັບຖານຂໍ້ມູນ

### ຄວາມໝາຍຂອງຖານຂໍ້ມູນ

Databases ຫຼື ຖານຂໍ້ມູນຄືກຸ່ມຂອງຂໍ້ມູນທີ່ຖືກເກັບລວບລວມໄວ້ໂດຍມີຄວາມສໍາພັນເຊິ່ງກັນ ແລະ ກັນໂດຍບໍ່ໄດ້ບັງຄັບວ່າຂໍ້ມູນທັງຫມົດນີ້ຈະຕ້ອງເກັບໄວ້ໃນແຟ້ມຂໍ້ມູນດຽວກັນ ຫຼື ແຍກເກັບຫຼາຍໆແຟ້ມຂໍ້ມູນ

ລະບົບຖານຂໍ້ມູນຄືລະບົບທີ່ລວບລວມຂໍ້ມູນຕ່າງໆທີ່ກ່ຽວຂ້ອງກັນເຂົ້າໄວ້ດ້ວຍກັນຢ່າງມີລະບົບ,ມີຄວາມສໍາພັນລະຫວ່າງຂໍ້ມູນຕ່າງໆທີ່ຊັດເຈັນໃນລະບົບຖານຂໍ້ມູນຈະປະກອບດ້ວຍແຟ້ມຂໍ້ມູນຫຼາຍແພ້ມທີ່ມີຂໍ້ມູນກ່ຽວຂ້ອງກັນ,ສໍາພັນກັນເຂົ້າໄວ້ດ້ວຍກັນຢ່າງເປັນລະບົບ ແລະ ເປີດໂອກາດໃຫ້ ຜູ້ໃຊ້ສາມາໃຊ້ງານ ແລະ ຮັກສາປ້ອງກັນຂໍ້ມູນເຫຼົ່ານີ້ໄດ້ຢ່າງມີປະສິດທິພາບໂດຍມີຊອບແວຣ໌ທີ່ປຽບສະເຫມືອນຊື່ກາງລະຫວ່າງຜູ້ໃຊ້ ແລະໂປຣແກຣມຕ່າງໆທີ່ກ່ຽວຂ້ອງກັບການໃຊ້ຖານຂໍ້ມູນເອີ້ນວ່າລະບົບຈັດການຖານຂໍ້ມູນ ຫຼື DBMS (Databases Management System)ມີຫນ້າທີ່ຊ່ວຍໃຫ້ຜູ້ໃຊ້ເຂົ້າເຖິງຂໍ້ມູນໄດ້ງ່າຍສະດວກ ແລະ ມີປະສິດທິພາບການເຂົ້າເຖິງຂໍ້ມູນຂອງຜູ້ໃຊ້ອາດເປັນການສ້າງຖານ ຂໍ້ມູນ,ການແກ້ໄຂຖານຂໍ້ມູນ ຫຼື ການຕັ້ງຄໍາຖາມເພື່ອໃຫ້ໄດ້ຂໍ້ມູນມາໂດຍຜູ້ໃຊ້ບໍ່ຈໍາເປັນຕ້ອງຮັບຮູ້ກ່ຽວກັບລາຍລະອາດພາຍໃນໂຄງສ້າງຂອງຖານຂໍ້ມູນ.

### ຄຸນລັກສະນະຂອງຖານຂໍ້ມູນ

* ລົດຄວາມຊ້ຳຊ້ອນຂອງຖານຂໍ້ມູນໃຫ້ເຫຼືອນ້ອຍທີ່ສຸດ (Minimum Redundancy)
* ມີຄວາມຖືກຕ້ອງສູງສຸດ (Maximum Integrity)
* ມີຄວາມເປັນອິດສະລະຂອງຂໍ້ມູນ (Data Independence)
* ມີລະບົບຄວາມປອດໄພຂອງຂໍ້ມູນສູງ (High Degree of Data Security)
* ການຄວບຄຸມຖານຂໍ້ມູນຈະຢູ່ສ່ວນກາງ (Logically Centralized Control)
  + - 1. **ຄວາມສໍາຄັນຂອງລະບົບຖານຂໍ້ມູນ**
* ສາມາດລົດຄວາມຊ້ຳຊ້ອນຂອງຂໍ້ມູນໄດ້
* ຫຼີກລ້ຽງຄວາມຂັດແຍ້ງຂອງຂໍ້ມູນໄດ້
* ສາມາດໃຊ້ຂໍ້ມູນຮ່ວມກັນໄດ້
* ສາມາດກໍານົດຄວາມເປັນມາດຕະຖານດຽວກັນຂອງຂໍ້ມູນໄດ້
* ສາມາດກໍານົດລະບົບຄວາມປອດໄພຂອງຂໍ້ມູນໄດ້
* ສາມາດເກີດຄວາມເປັນອິດສະຫຼະຂອງຂໍ້ມູນໄດ້

### ສະຖາປັດຕະຍະກໍາລະບົບຖານຂໍ້ມູນ

ສະຖາປັດຕະຍະກໍາລະບົບຖານຂໍ້ມູນ ມີ 3 ລະດັບຄື:

* ລະດັບພາຍໃນ (Internal Level)
* ລະດັບລັກສະນະແນວຄິດ (Conceptual Level)
* ລະດັບພາຍນອກ (External Level)

1. ລະດັບພາຍໃນ (Internal Level)

ເປັນການນໍາເອົາຂໍ້ມູນທີ່ໄດ້ຈາກລະດັບແນວຄິດມາວິເຄາະ ແລະ ອອກແບບໂດຍແປງໃຫ້ຢູ່ໃນຮູບແບບຂອງການຈັດການຖານຂໍ້ມູນ (DBMS)ທີ່ເລືອກໃຊ້ໂດຍເສີມແນວ ຄິດການເຮັດ Normalization car Demoralization

1. ລະດັບລັກສະນະແນວຄິດ(Conceptual Level)

ເປັນການນໍາເອົາຂໍ້ມູນທີ່ໄດ້ຈາກການວິເຄາະຄວາມຕ້ອງການຂອງຜູ້ໃຊ້ຂໍ້ມູນໃນລະດັບພາຍນອກມາອອກແບບຖານຂໍ້ມູນເພື່ອໄດ້ໂຄງຮ່າງຂອງຖານຂໍ້ມູນໃນລະດັບແນວຄິດທີ່ປະກອບດ້ວຍໂຄງສ້າງຂອງຖານຂໍ້ມູນສ່ວນຫນຶ່ງເປັນຄວາມສໍາພັນກັນ.

1. ລະດັບພາຍນອກ (External Level)

ເປັນການນໍາເອົາຂໍ້ມູນທີ່ໄດ້ຈາກລະດັບພາຍໃນມາກໍານົດໂຄງສ້າງຂໍ້ມູນ ແລະ ການຈັດເກັບວິທີການເຂົ້າເຖິງການຈັດການດ້ານລະບົບຄວາມປອດໄພຂອງຂໍ້ມູນເພື່ອຖານຂໍ້ມູນເຮັດວຽກໄດ້ຢ່າງມີປະສິດທິພາບ.

* + 1. **ຄວາມໝາຍ ແລະ ຄວາມສຳຄັນຂອງການອອກແບບລະບົບ**
    2. **ທິດສະດີທີ່ກ່ຽວຂ້ອງກັບ DFD (Data Flow Diagram)** 
       1. **ຈຸດປະສົງຂອງ DFD**
* ເປັນແຜນພາບທີ່ສະຫຼຸບລວມຂໍ້ມູນທັງໝົດໄດ້ຈາກການວິເຄາະໃນລັກສະນະຂອງຮູບແບບທີ່ເປັນໂຄງສ້າງ.
* ເປັນຂໍ້ຕົກລົງຮ່ວມກັນລະຫວ່າງນັກວິເຄາະລະບົບ ແລະ ຜູ້ຊົມໃຊ້.
* ເປັນແຜນພາບທີ່ໃຊ້ໃນການພັດທະນາຕໍ່ໃນຂັ້ນຕອນຂອງການອອກແບບ.
* ຮູ້ທີ່ໄປທີ່ມາຂອງຂໍ້ມູນທີ່ໄຫຼໃນຂະບວນການຕ່າງໆ.
  + - 1. **ສັນຍະລັກທີ່ໃຊ້ໃນແຜນວາດການໄຫຼຂໍ້ມູນ**

|  |  |  |
| --- | --- | --- |
| **ຊື່** | **ສັນຍາລັກ** | **ຄວາມຫມາຍ** |
| Boundary Or External Entity |  | ຂອບເຂດຫມາຍເຖິງພາກ ສ່ວນທີ່ກ່ຽວຂ້ອງເກັບລະບົບເຊິ່ງລະບົບບໍ່ສາມາດຄວບຄຸມໄດ້ |
| Process |  | ປະມວນຜົນຫຼືຫນ້າວຽກທີ່ເຮັດໃນໂຄງການນັ້ນໆ |
| Data Store |  | ບ່ອນຈັດເກັບຂໍ້ມູນ |
| Data Flow |  | ການໄຫຼຂອງຂໍ້ມູນ |
| Real-Time Link |  | ການເຊື່ອມໂຍງແບບໄກທີ່ມີການຕອບກັບແບບທັນທີທັນໃດ |

ຕາຕະລາງທີ 2 ສະແດງສັນຍາລັກ **Data Flow Diagram**

* + - 1. **ກົດຂອງ Process**
* ຕ້ອງບໍ່ມີຂໍ້ມູນເຂົ້າພຽງຢ່າງດຽວ
* ຕ້ອງບໍ່ມີຂໍ້ມູນອອກພຽງຢ່າງດຽວ
* ຂໍ້ມູນທີ່ສົ່ງເຂົ້າຕ້ອງພຽງພໍໃນການສ້າງຖານຂໍ້ມູນທີ່ສົ່ງອອກ
* ການຕັ້ງຊື່ Process ຕ້ອງໃຊ້ຄໍາກິລິຍາ(Verb) ເຊັ່ນ: ບັນທຶກຂໍ້ມູນ,ໃບບິນ,ກວດສອບຂໍ້ມູນລູກຄ້າ,ຈໍານວນເງິນເດືອນ.

## ຫຼັກການຂຽນແຜນວາດການໄຫຼຂໍ້ມູນ

1. Process:
   * ເມື່ອມີຂໍ້ມູນເຂົ້າໄປທີ່ Processກໍ່ຕ້ອງມີຂໍ້ມູນ ຫຼື ຜົນຮັບອອກມາຈາກ Processເຊັ່ນກັນຈະເປັນໄປບໍ່ໄດ້ທີ່ມີສະເພາະຂໍ້ມູນເຂົາຢ່າງດຽວ.
2. Data store:

* ຂໍ້ມູນຈະໄຫຼຈາກ Data storeໜຶ່ງໄປຫາ Data storeໜຶ່ງໂດຍກົງບໍ່ໄດ້ຈະຕ້ອງຜ່ານProcessເທົ່ານັ້ນ
* ຂໍ້ມູນທີ່ສົ່ງຜ່ານ External entityບໍ່ສາມາດໄຫຼເຂົ້າໄປ Data storeໂດຍກົງໄດ້ຈະຕ້ອງໃຊ້ Processເປັນຕົວກາງໃນການເຊື່ອມໂຍງເພື່ອຈັດເກັບຂໍ້ມູນໃນ Data store
* ຂໍ້ມູນທີ່ໄຫຼຜ່ານຈາກ Data storeບໍ່ສາມາດເຊື່ອໂຍງເຂົ້າກັບ External entityໄດ້ໂດຍກົງຈະຕ້ອງຜ່ານ Processເທົ່ານັ້ນ.

1. External entity:

* External entityບໍ່ສາມາດເຊື່ອມໂຍງເຂົ້າຫາກັນໄດ້ຈະຕ້ອງໃຊ້ Processເປັນຕົວກາງເພື່ອສົ່ງຜ່ານ ແລະ ຊື່ຂອງ External entityຈະໃຊ້ຄໍານາມເທົ່ານັ້ນ.

1. Data flow:

* ການໄຫຼຂໍ້ມູນທີ່ມີຫົວຊີ້ໄປທີ່ Processໝາຍເຖິງ Processມີການອ່ານ ຫຼື ການດືງຂໍ້ມູນຈາກ Data storeມາໃຊ້ວຽກ
* ການໄຫຼຂໍ້ມູນຈາກ Process ທີ່ມີຫົວລູກສອນຊີ້ໄປຍັງData storeໝາຍເຖິງການ Updateຫຼື ການເພີ່ມຂໍ້ມູນລົງໄປທີ່ Data store
* ການໄຫຼຂໍ້ມູນທີ່ມີຫົວລູກສອນທັງສອງດ້ານທີ່ເຊື່ອມໂຍງລະຫວ່າງ Processກັບ Data store ໝາຍເຖິງມີການດືງຂໍ້ມູນຈາກ Data storeມາປັບປຸງ ແລະ ມີການUpdateຂໍ້ມູນລົງໄປໃນ Data store
* ການໄຫຼຂໍ້ມູນບໍ່ສາມາດຢ້ອນກັບໄປຍັງ Processເດີມໄດ້ ຢ່າງໝ່ອຍຕ້ອງເຊື່ອມໂຍງຜ່ານProcessໜື່ງເພື່ອສົ່ງຜ່ານຢ້ອນກັບມາຍັງ Processເດີມ ແລະ ຊື່ທີ່ລະບຸໃນການໄຫຼຂໍ້ມູນຈະໃຊ້ຄໍານາມ

|  |  |
| --- | --- |
| **ອະນຸຍາດ** | **ບໍ່ອະນຸຍາດ** |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

ຕາຕະລາງທີ 10 ຮູບການປຽບທຽບແຜນວາດການໄຫຼຂໍ້ມູນທີ່ບໍ່ຖືກຕ້ອງ ແລະ ຖືກຕ້ອງ

* + 1. **ຄວາມໝາຍ ແລະ ສັນຍະລັກຂອງ Flowchart**
    2. **ແຜນວາດຄວາມສຳພັນຂອງ Entity (ER Diagram)**

ການອອກແບບການສ້າງຕາຕະລາງຖານຂໍ້ມູນແບບຈໍາລອງ ERແບ່ງອອກເປັນ 2 ຂັ້ນຕອນຫຼັກ ຄື: ຂັນຕອນທໍາອິດເປັນການສ້າງແບບຈໍາລອງ ER ຂັ້ນຕອນນີ້ຈະກ່າວເຖີງຄວາມໝາຍຂອງສັນຍາລັກຕ່າງໆທີ່ໃຊ້ໃນແບບຈໍາລອງ ER, ຂັ້ນຕອນທີ່ 2 ແມ່ນການແປງແບບຈໍາລອງ ER ໃຫ້ເປັນຕາຕະລາງຂໍ້ມູນເພື່ອໃຊ້ເປັນຕົວຈັດການຂໍ້ມູນ.ເມື່ອເຂົ້າໃຈໃນທັງສອງແບບນີ້ແລ້ວສາມາດອອກແບບຖານຂໍ້ມູນ ແລະ ສ້າງຖານຂໍ້ມູນໃນແບບຈໍາລອງ ER ຂຶ້ນມາໃຊ້ງານດ້ວຍຕົນເອງ.

### ສັນຍາລັກທີ່ໃຊ້ໃນແບບຈໍາລອງ ER

ການອອກແບບຈໍາລອງຖານຂໍ້ມູນ ER ນັ້ນກ່ອນອື່ນຕ້ອງຮູ້ຈັກຄວາມໝາຍສັນຍາລັກຕ່າງໆທີ່ໃຊ້ໃນການອອກແບບຈໍາລອງ ER ໄດ້ຢ່າງຖືກຕ້ອງດັ່ງນັ້ນໃນຫົວຂໍ້ນີ້ຈະເວົ້າເຖິງຄວາມໝາຍ ແລະ ການໃຊ້ງານສັນຍາລັກຕ່າງໆຂອງ ER.

* 1. ເອັນຕີຕີ້ (Entity)

ເອັນຕີຕີ້ຄືວັດຖຸທີ່ເຮົາສົນໃຈເຊິ່ງອາດເປັນໄດ້ທັງບຸກຄົນ,ສະຖານທີ່,ວັດຖຸ,ເຫດການ ຫຼືແນວຄິດທີ່ກໍ່ໃຫ້ເກີດກຸ່ມຂອງຂໍ້ມູນທີ່ຕ້ອງການເອັນຕີຕີ້ແບ່ງອອກເປັນ 2 ປະເພດຄື:

* 1. Strong Entity:

ເປັນເອັນຕີຕີ້ທີ່ເກີດຂຶ້ນດ້ວຍຕົນເອງເປັນອິດສະຫຼະບໍ່ຂຶ້ນກັບເອັນຕີຕີໃດສັນຍາລັກທີ່ໃຊ້ຄືຮູບສີ່ຫຼ່ຽມ ແລະ ສາມາດເອີ້ນ Strong Entity ໄດ້ອີກຊື່ໜື່ງວ່າ Regular Entity.

**CUSTOMER**

**STOCK**

ຮູບທີ 3 ຮູບStrong Entity

* 1. Weak Entity:

ເອັນຕີຕີ້ຊະນິດນີ້ຈະຂຶ້ນກັບເອັນຕີຕີ້ຊະນິດອື່ນໆບໍ່ສາມາດເກີດຂຶ້ນໄດ້ຕາມລໍາພັງ ແລະ ຈະຖືກລົບເມື່ອເອັນຕີຕີ້ຫຼັກຖືກລົບອອກ ສັນຍາລັກທີ່ໃຊ້ຄືຮູບສີ່ຫຼ່ຽມຊ້ອນກັນ.

**CUSTOMER**

**STOCK**

ຮູບທີ 4 ຮູບWeak Entity

* 1. ແອັດທີຣບີວ(Attribute)

ແອັດທີຣບີວຄືຄຸນສົມບັດຂອງສົມບັດຂອງເອັນຕີຕີ້ສັນຍາລັກຂອງແອັດທີຣບີວຈະເປັນຮູບວົງຮີແອັດທີຣບີວໃດທີ່ຖືກໃຊ້ເປັນຄີຫຼັກຈະຖືກຂີດເສັ້ນກ້ອງກໍາກັບໄວ້.

emName

Position

emNo

**EMPLOYEE**

ຮູບທີ 5 ຮູບAttribute

* 1. ຄວາມສໍາພັນ (Relation)

ຄວາມສໍາພັນໃນທີ່ນີ້ໝາຍເຖິງຄວາມສໍາພັນລະຫວ່າງເອັນຕີຕີ້ໂດຍແຕ່ລະຄວາມສໍາພັນຄວນມີຊື່ລະບຸໄວ້ເພື່ອໃຊ້ອະທິບາຍເຊິ່ງປົກກະຕິຈະໃຊ້ສັນຍາລັກຮູບດອກຈັນທີ່ພາຍໃນລະບຸຄໍາກິລິຍາໄວ້ເພື່ອອະທິບາຍຄວາມສໍາພັນ.

SUBJECT

teaches

LUCTURE

M

1

ຮູບທີ 6 ຮູບ Relation

## ຄວາມຮູ້ພື້ນຖານກ່ຽວກັບພາສາ NOSQL

ຫຼາຍ​ຄົນ​ຄົງ​ພໍ​ຈະ​ເຄີຍ​ໄດ້ຍິນ​ກັນ​ມາແດ່ກ່ຽວກັບ​ເທັກ​ໂນ​ໂລ​ຢີ​ການ​ຈັດການ​ຂໍ້​ມູນ​ແບບ​ໃໝ່​ນີ້ ຊຶ່ງ​ກໍ​ຄື NoSQL ເມື່ອ​ເວົ້າ​ເຖິງ NoSQL ຈະ​ໄດ້ຍິນ​ຊື່​ເວັບ​ໄຊ​​ທີ່​ໃຫ່ຍ​ໆ ເຊັ່ນ Facebook, Twitter, FourSquare, Digg ແລະ ​ອື່ນ​ໆ.  ເຮັດໃຫ້​ເຮົາ​ຮັບ​ຮູ້​ວ່າ NoSQL ເປັນ​ລະບົບ​ຖານ​ຂໍ້​ມູນ​ສຳລັບ​ງານ​ທີ່​ຕ້ອງ​ຮອງ​ຮັບ​ຂໍ້​ມູນ​ຂະໜາດ​ໃຫ່ຍໆຮອງ​ຮັບ​ການ​ຂະຫຍາຍ​ລະບົບ​ໄດ້​ງ່າຍເປັນ​ຕົ້ນ.

ຊຶ່ງ​ກໍ​ເປັນ​ເຊັ່ນ​ນັ້ນ​ແທ້ ແຕ່​ງານ​ທີ່​ນ້ອຍ​ໆຈະ​ເຮັດ​ຢ່າງໃດໃຊ້​ງານ​ໄດ້​ບໍ່ ຄຸ້ມ​ຄ່າ​ທີ່​ຈະ​ນຳ NoSQL ມາ​ໃຊ້​ງານ​ຂະໜາດ​ນ້ອຍ​ ຫຼື ບໍ່ ຫຼື​ໃຊ້ Relational Database ກໍ​ພຽງ​ພໍ​ແລ້ວ ຄຳ​ຕອບ​ຄືຂຶ້ນ​ຢູ່​ກັບ​ລັກສະນະ​ໃນ​ການ​ນຳ​ມາ​ໃຊ້​ງານ ກ່ອນທີ່ຈະ​ຕອບ​ຄຳ​ຖາມວ່າ NoSQL ເປັນ​ຄຳ​ຕອບ​ຂອງ​ລະບົບ​ຈັດ​ເກັບ​ຂໍ້​ມູນ​ຂອງ​ຫຼືບໍ່ ລອງ​ພິຈາລະນາ​ຫົວຂໍ້​ຕ່າງ​ໆດັ່ງ​ຕໍ່ໄປ​ນີ້:

1. ຜູ້ໃຊ້ນັບມື້ນັບຫຼາຍ (BigUsers)

ຈະ​ເຫັນ​ໄດ້​ວ່າ​ໃນ​ຊ່ວງ​ເວລາ​ທີ່​ຜ່ານມາ​ນີ້ ແລະ ​ໃນ​ປັດ​ຈຸ​ບັນຜູ້​ທີ່​ໃຊ້​ງານ Internet ມີ​ແນວ​ໂນ້ມ​ຫລາຍ​ຂຶ້ນ​ເລື້ອຍ​ໆ ບໍ່​ວ່າ​ຈະ​ໃຊ້​ງານ​ຜ່ານ Desktop PC ຫລື Smartphone ຊຶ່ງ​ເທັກ​ໂນ​ໂລ​ຊີ​ຂອງ​ອຸປະ ກອນ (Devices) ມີ​​ຫຼາຍ​​ຂຶ້ນ ແລະ ​ໃຊ້​ງານ​ໄດ້​ງ່າຍ​ຂຶ້ນ

ການ​ພັດທະນາ​ລະບົບ​ໃຫ້​ສາມາດ​ຮອງ​ຮັບ​ປະລິມານ​ການ​ເຂົ້າ​ໃຊ້​ງານ​ແຕ່​ລະ​ອຸປະກອນ (Devices) ເປັນ​ສິ່ງ​ໜຶ່ງ​ທີ່​ຕ້ອງ​ນຳ​ມາ​ພິຈາລະນາ  ແລະ ​ບໍ່​ພຽງ​ແຕ່​ຕ້ອງ​ຮອງ​ຮັບ​ການ​ເຂົ້າ​ໃຊ້​ງານ​ຂອງ​ຜູ້​ໃຊ້​ໄດ້​ເທົ່າ​ນັ້ນ  ເຮົາ​ຕ້ອງ​ຮອງ​ຮັບ​ວິທີ​ການ​ປ້ອນ​ຂໍ້​ມູນ​ແບບ​ໃໝ່ຄືແຕ່​ກ່ອນ​ຜູ້​ຈັດການ​ເນື້ອ​ຫາ​ຕ່າງ​ໆ ຄື Web Master, Web Editor, ຜູ້​ເບິ່ງ​ແຍງ​ລະບົບເປັນ​ຕົ້ນ ແຕ່​ປະຈຸບັນຜູ້​ທີ່​ປ້ອນ​ຂໍ້​ມູນ​ຄືຜູ້​ໃຊ້​ບໍລິການ (users) ໂດຍ​ກົງຜ່ານ​ອຸປະກອນ (Devices) ຕ່າງ​ໆ​ທີ່​ມີ​ຫຼາກ​ຫຼາຍ ແລະ ​ການ​ປ້ອນ​ຂໍ້​ມູນ​ກໍ​ງ່າຍ​ກວ່າ​ແຕ່​ກ່ອນ​ອີກດ້ວຍ

ຍັງມີ​ປັດ​ໄຈ​ອື່ນ​ໆເຊັ່ນ ເທດ​ສະການ​ສຳຄັນ​ໆທີ່​ຄົນ​ຈະ​ເຂົ້າ​ມາ​ໃຊ້​ງານ​ຫລາຍ​ເປັນ​ພິເສດ ຫຼື ​ຜູ້​ໃຊ້​ງານ​ທີ່​ບໍ່​ແມ່ນ​ແຕ່​ປະເທດ​ເຮົາ​ເທົ່າ​ນັ້ນ ເພາະ​ໂລກ​ອິນ​ເຕີເນັດ​ເຖິງ​ກັນ ອາດຈະ​ຕ້ອງ​ເບິ່ງ​ວ່າ​ລະບົບ​ເຮົາ​ມີ​ຜູ້​ເຂົ້າ​ໃຊ້​ງານ​ຈາກ​ຕ່າງ​ປະເທດ ຫຼື ທົ່ວ​ໂລກ ​ຫຼື ບໍ່​ອີກ​ດ້ວຍ

ດັ່ງ​ນັ້ນ ​ເຮົາ​ຕ້ອງ​ກັບ​ມາ​​ທົບທວນ​ວິທີ​ການ​ຈັດການ​ຖານ​ຂໍ້​ມູນ​ແລ້ວ​ວ່າ  ບໍ່​ແມ່ນ​ແຕ່​ເຮັດໃຫ້​ຮອງ​ຮັບ​ກັບ​ການ​ເຂົ້າ​ມາ​ໃຊ້​ງານ​ຂອງ​ຜູ້​ໃຊ້​ບໍລິການ​ເທົ່າ​ນັ້ນ ແຕ່​ຕ້ອງ​ຮອງ​ຮັບ​ການ​ຈັດ​ເກັບ​ຂໍ້​ມູນ​ທີ່​ຫລາຍ​ຂຶ້ນ​ເລື້ອຍ​ໆໄດ້​ອີກ

1. ປະເພດຂໍ້ມູນຕ່າງໆ ແລະ ຂໍ້ມູນທີ່ຕ້ອງການຈັດເກັບຫຼາຍຂຶ້ນເລື້ອຍໆ (BigData)

  ຈາກ​ຕົວ​ແປ​ຂອງ​ຜູ້​ໃຊ້​ງານ​ມີ​ຫລາຍ​ຂຶ້ນ​ເລື້ອຍ​ໆ ອຸປະກອນ​ໃນ​ການ​ເຂົ້າ​ໃຊ້​ງານ​ກໍ​ຫຼາກ​ຫຼາຍປະ​ເພດ​ຂອງ​ຂໍ້​ມູນ​ທີ່​ໄດ້​ຈາກ​ແຕ່​ລະ​ອຸປະກອນ​ກໍ​ຫຼາກ​ຫຼາຍ​ປະ​ເພດ ເຊັ່ນ ຂໍ້ຄວາມ, ຮູບ​ພາບ, ສຽງ, ວິ​ດີ​ໂອ, ຕຳແໜ່ງ​ສະຖານ​ທີ່ (GeoLocation) ແລະ ​ອື່ນ​ໆ ແລະ ​ການ​ປ້ອນ​ຂໍ້​ມູນ​ເຫຼົ່າ​ນີ້​ກໍ​ງ່າຍ​ແສນ​ງ່າຍ ເພາະ​ເທັກ​ໂນ​ໂລ​ຢີ​ຂອງ​ຮາ​ດ​ແວ​ຣ ແລະ ​ຊອບແວ​​ມີ​ການ​ພັດທະນາ​ຫລາຍ​ຂຶ້ນ​ເລື້ອຍ​ໆ ໃຊ້​ງານ​ງ່າຍ​ຂຶ້ນສະ​ດວກ​ຂຶ້ນວ່ອງໄວ​ຂຶ້ນ​ເລື້ອຍ​ໆ

    ດັ່ງ​ນັ້ນ​ການ​ຈັດ​ເກັບ​ຂໍ້​ມູນ​ທີ່​ລັ່ງໄຫຼ​ເຂົ້າ​ມາ​ຈາກ​ອຸປະກອນ​ຕ່າງ​ໆ​ເຫຼົ່າ​ນີ້ ເຮົາ​ອາດຈະ​ຕ້ອງ​ນຳ​ມາ​​ວິ​ເຄາະ​ພຶດຕິກຳ​ຂອງ​ຜູ້​ໃຊ້​ບໍລິການ ການ​ສົ່ງ​ເສີມ​ກາ​ນຕະຫຼາດ ເຮັດ​ຂໍ້​ມູນ​ການ​ຕັດ​ສິນ​ໃຈ​ຂອງ​ຜູ້​ບໍລິຫານ ຂໍ້​ມູນ​ລູກ​ຄ້າ​ສຳພັນ ແລະ ​ອື່ນ​ໆ​ອີກ​ຫລວງ​ຫລາຍ  ການ​ນຳ​ລະບົບ​ຖານ​ຂໍ້​ມູນ​ແບບ​ເດີມ (Relational Database) ອາດຈະ​ບໍ່​ເໝາະ​ກັບ​ລັກສະນະ​ງານ​ບາງຢ່າງ​ອີກ​ຕໍ່ໄປ

1. ເຕັກໂນໂລຢີຮາດແວໄດ້ປ່ຽນແປງລາຄາກໍ່ຖືກຫຼຸດລົງແຕ່ວ່າປະສິດທິພາບດີຂື້ນ (Cloud Computing).

ເຮົາ​ອາດຈະ​ເລີ່ມ​ໄດ້ຍິນ​ຄຳ​ວ່າ Cloud Technology  ຊຶ່ງ​ກໍ​ມີ​ຫຼາຍ​ປະ​ເພດແຕ່​ໃນ​ທີ່​ນີ້ເຮົາ​ຈະ​ເວົ້າ​ເຖິງ​ໃນ​ແງ່​ຂອງ​ການ​ນຳ​ມາ​ໃຊ້​ງານໂດຍ​ປະຈຸບັນ​ຖ້າ​ໃຜ​ເຄີຍ​ໄດ້​ລອງ​ໃຊ້​ງານ EC2 ຂອງ Amazon ມາແດ່​ແລ້ວຈະ​ຮູ້​ຈັກ​ເປັນ​ຢ່າງ​ດີ​ວ່າການ​ຈະ​ມີ​ເຄື່ອງ Server ແຮງ​ໆ​ຈັກເຄື່ອງ ເປັນ​ເລື່ອງ​ທີ່​ງ່າຍ​ຫລາຍ​ ຫຼື ການ​ຈະ​ມີ Server 10 ເຄື່ອງ ຫລື 20 ເຄື່ອງ ນຳ​ມາ​ຕໍ່​ເປັນ Database Cluster ນັ້ນ​ງ່າຍ​ຫຼາຍ​ພຽງແຕ່​ຄລິກ​ສ້າງ Instance ບໍ່​ຈັກ​ເທື່ອ​ກໍ​ໄດ້ Server ມາ​ໃຊ້​ງານ​ແລ້ວ ແລະ​ ລາ​ຄາ​ກໍ​ຖືກ​ຫລາຍ​ໆ ຖ້າ​ເຮົາ​ບໍ່​ໃຊ້​ງານ​ແລ້ວກໍ​ຍົກ​ເລີກ​ການ​ໃຊ້​ງານ ​ແລະ​ ຄືນ​ກັບ​ໄປ​ໄດ້​ທັນ​ທີ ເມື່ອ​ທຽບ​ກັບ​ສະໄໝ​ກ່ອນ​ທີ່​ເຮົາ​ຕ້ອງ​ການ​ມີ Server ຈັກ 10 ເຄື່ອງ ຈະ​ຕ້ອງ​ລົງ​ທຶນ​ຊື້​ເຄື່ອງ​ມາ​ຫລາຍ ຖ້າ​ເຊົາ​ໃຊ້​ງານ​ແລ້ວຕ້ອງ​ແບກ​ຮັບ​ພາລະ​ເຄື່ອງ Server ເຫຼົ່າ​ນີ້​ໄວ້​ຊຶ່ງ​ເປັນ​ຕົ້ນ​ທຶນ​ທີ່​ແພງ​ຫລາຍ​

ຈາກ​ທີ່​ກ່າວ​ມາ​ຂ້າງ​ເທິງ ເຮົາ​ບໍ່​ໄດ້​ເນັ້ນ​ຂໍ້​ດີ​ຂອງ EC2 ແຕ່​ຢ່າງ​ໃດແຕ່​ກຳ​ລັງ​ຈະ​ເນັ້ນ​ວ່າພາບ ລວມ​ຂອງ​ການ​ໃຊ້​ງານ Server ເລີ່ມ​ປ່ຽນ​ໄປຄືໃຊ້​ງານ​ໄດ້​ງ່າຍ​ຂຶ້ນ ລາ​ຄາ​ຖືກ​ລົງ ແຕ່​ປະ​ສິດ​ທິ​ພາບ​ດີ​ຂຶ້ນ ຊຶ່ງ​ເປັນ​ສິ່ງ​ທີ່​ສຳຄັນ​ໃນ​ການ​ນຳ​ມາ​ພິຈາລະນາ​ຂອງ​ເທກ​ໂນ​ໂລ​ຢີ​ດ້ານ​ຖານ​ຂໍ້​ມູນ​ຄື ຖ້າ​ຕ້ອງ​ການ​ຈັດ​ເກັບ​ຖານ​ຂໍ້​ມູນ​ໃຫຍ່​ໆ ຫຼື ຮອງ​ຮັບ​ຜູ້​ໃຊ້​ງານ​ໄດ້​ເປັນ​ຈຳນວນ​ຫລາຍ​ໆ ການ​ຂະຫຍາຍ​ລະບົບ​ຖານ​ຂໍ້​ມູນ​ເປັນ​ເລື່ອງ​ທີ່​ງ່າຍ​ຂຶ້ນ ຊຶ່ງ​ເຮັດ​ໄດ້​ໂດຍ​ການ​ເອົາ Server ມາ​ຕໍ່​ໆ​ກັນ​ອອກ​ໄປ ຫລື​ ເອີ້ນວ່າ​ການ​ຂະຫຍາຍ​ອອກ​ແນວ​ນອນ (Scale Out) ບໍ່​ແມ່ນ​ການ​ຂະຫຍາຍ​ລະບົບ​ຄືແຕ່​ກ່ອນ ຄື​ຂະຫຍາຍ​ອອກ​ແນວ​ຕັ້ງ (Scale Up) ແລະ​ ຕ້ອງ​ໃຊ້​ເຄື່ອງ Server ທີ່​ມີ​ປະ​ສິດ​ທິ​ພາບ​ສູງ​ໆ ຊຶ່ງ​ຈະ​ມີ​ຕົ້ນ​ທຶນ​ທີ່​ແພງ​ກວ່າ​ການ​ຂະຫຍາຍ​ແບບ​ແນວ​ນອນຫລາຍ​

     ດັ່ງ​ນັ້ນ ​ການ​ຂະຫຍາຍ​ລະບົບ​ທີ່​ຢູ່​ເທິງ​ພື້ນ​ຖານ​ຂອງ NoSQL ຄື​ຮອງ​ຮັບ​ການ​ຂະຫຍາຍ​ລະ ບົບ​ແບບ​ແນວ​ນອນ (Scale Out) ຊຶ່ງ​ຈະ​ກະ​ຈາຍ​ຂໍ້​ມູນ​ໄປ​ເກັບ​ທີ່​ເຄື່ອງ Server ຫຼາຍ​ໆ​ເຄື່ອງ ແລະ ​ໃຊ້​ເຄື່ອງ Server ທົ້ວ​ໆ​ໄປ​ທີ່​ເອີ້ນວ່າ (Commodity Server) ບໍ່​ຈຳ​ເປັນ​ຕ້ອງ​ໃຊ້ Server ທີ່​ເອີ້ນວ່າ Enterprise Server ທີ່​ມີ​ລາ​ຄາ​ແພງ​ຕາມ Spec ທີ່​ສູງ​ຂຶ້ນ​ເລື້ອຍ​ໆ ແລະ​ ການ​ບໍລິຫານ​ຈັດການ​ກໍ​ຍາກ​ຂຶ້ນ​ອີກ​ດ້ວຍ

1. ບັນຫາ​ຂອງ Relational Database

ຖ້າ​ໃຜ​ທີ່​ໃຊ້​ງານ Relation Database ທີ່​ຕ້ອງ​ການ​ຮອງ​ຮັບ​ການ​ຈັດ​ເກັບ​ຂໍ້​ມູນ​ຂະໜາດ​ໃຫຍ່​ໆ ຄົງ​ຫຼີກລ້ຽງບໍ່ໄດ້ເລື່ອງ​ການເຮັດ Sharding ແລະ Distributed Cache ເພາະ​ເປັນ​ຕັວ​ຫຼັກ​ທີ່​ຕ້ອງ​ເຮັດ ເພື່ອ​ຂະຫຍາຍ​ລະບົບ​ຖານ​ຂໍ້​ມູນ​ຂອງ Relational Database ໃຫ້​ສາມາດ​ຮອງ​ຮັບ​ຂໍ້​ມູນ​ທີ່​ຫລາຍ​ຂຶ້ນ ແລະ​ຮອງ​ຮັບ​ຈຳນວນ​ການ​ເຂົ້າ​ມາ​ໃຊ້​ງານ​ລະບົບ​ໄດ້​ຫຼາຍ​ຂຶ້ນ

1. Manual Sharding

    ການ​ແບ່ງ​ຕາ​ຕະລາງ​ຖານ​ຂໍ້​ມູນ (Table) ອອກ​ເປັນ​ສ່ວນ​ໆ ແລ້ວ​ກໍ່ການ​ກະ​ຈາຍ​ໄປ​ຈັດ​ເກັບ​ໃນ​ຫຼາຍ​ໆ Server  ເພື່ອ​ໃຫ້​ແຕ່​ລະ​ຕາ​ຕະລາງ (Table) ຂອງ​ຖານ​ຂໍ້​ມູນ​ບໍ່​ຈັດ​ເກັບ​ຂໍ້​ມູນ​ທີ່​ຫຼາຍ​ເກີນ​ໄປ ເພາະ​ຖ້າ​ຂໍ້​ມູນ​ໃນ​ແຕ່​ລະ​ຖານ​ຂໍ້​ມູນ​ຫູາຍ​ເກີນ​ໄປ ຈະ​ເຮັດໃຫ້​ລະບົບ​ຖານ​ຂໍ້​ມູນ​ຊ້າ​ ແຕ່​ບັນຫາ​ກໍ​ຈະ​ຕາມ​ມາ​ອີກ​ຄື ເມື່ອ​ຕ້ອງ​ກະ​ຈາຍ​ຂໍ້​ມູນ​ອອກ​ໄປ​ໃນ​ແຕ່​ລະ Server ການ​ຈະ​ ​ກັບ​ຂໍ້​ມູນ ເຊັ່ນ ເພີ່ມ, ແກ້​ໄຂ, ລົບ, ດຶງ​ຂໍ້​ມູນ​ມາ​ສະແດງ​ຕ່າງ​ໆ ຈະ​ຕ້ອງ​ເຮັດ​ຜ່ານ Application ຫຼື ​ຕ້ອງ​ມີ Server ບາງ​ໂຕ​ທີ່​​ຖ້າ​ດຶງ​ຂໍ້​ມູນ​ແຕ່​ລະ Server ມາ​ທັງໝົດ​ເປັນ​ກ້ອນ​ດຽວ ນັ້ນ​ໝາຍ​ຄວາມ​ວ່າ ເຮົາ​ຕ້ອງ​ເຮັດ​ດ້ວຍ​ໂຕ​ເຮົາ​ເອງ ບໍ່​ແມ່ນ​ລະບົບ​ຖານ​ຂໍ້​ມູນ​ຈັດການ​ໃຫ້ (Manual Sharding)

1. Distributed Cache

ເມື່ອ​ເຮົາ​ຕ້ອງ​ການ​ໃຫ້​ລະບົບ​ຮອງ​ຮັບ​ການ​ເຂົ້າ​ມາ​ໃຊ້​ງານ​ຫລາຍ​ໆ​ໄດ້​ນັ້ນ ຖ້າ​ຈະ​ຕ້ອງ​ເຂົ້າ​ມາ​ອ່ານ​ຂໍ້​ມູນ​ຜ່ານ Database ໂດຍ​ກົງ​ມັນອາດຈະ​ຮອງ​ຮັບ​ບໍ່​ໄຫວ ຫຼື ​ເຮັດ​ໄດ້​ຊ້າ ດັ່ງ​ນັ້ນ​ຈະ​ຕ້ອງ​ມີ​ການເຮັດ Cache Layer ຂຶ້ນ​ມາ ຄື​ແທນ​ທີ່​ຈະ​ເຂົ້າໄປ​ອ່ານ​ຈາກ​ຖານ​ຂໍ້​ມູນ​ໂດຍ​ກົງ ກໍ​ໃຫ້​ອ່ານ​ຜ່ານ Cache ກ່ອນ ດັ່ງ​ນັ້ນ​ການ​ອ່ານ​ຂໍ້​ມູນ​ຈາກ Cache ເປັນ​ການ​ອ່ານ​ຈາກ Memory ໂດຍ​ກົງ ເຮັດໃຫ້​ຮອງ​ຮັບ​ປະລິມານ​ການ​ເຂົ້າ​ມາ​ໃຊ້​ງານ​ໄດ້​ຫຼາຍ​ຂຶ້ນ

    ແຕ່​ບັນຫາ​ຄືການເຮັດ Cache Layer ນີ້ຮອງ​ຮັບ​ສະເພາະ​ການ​ອ່ານ​ຂໍ້​ມູນ​ເທົ່າ​ນັ້ນ ບໍ່​ຮອງ​ຮັບ​ການ​ຂຽນ​ຂໍ້​ມູນ​ໄດ້ ຖ້າ​ຕ້ອງ​ການ​ຮອງ​ຮັບ​ການ​ຂຽນ​ຂໍ້​ມູນ​ປະລິມານ​ຫຼາຍ​ໆ ແລະ​ ອ່ານ​ຂໍ້​ມູນ​ປະລິມານ​ຫຼາຍ​ໆ ຈຶ່ງ​ເປັນ​ສິ່ງ​ທີ່ Relational Database ບໍ່​ສາມາດ​ຮອງ​ຮັບ​ງານ​ໃນ​ລັກສະນະ ອ່ານ​,​ຂຽນ ຂໍ້​ມູນ​ປະລິມານ​ຫຼາຍ​ໆໄດ້​ດີ​ ແລະ​ ສິ່ງ​ສຳຄັນການເຮັດ Cache Layer ຈະ​ຕ້ອງ​ມີ​ການ​ດູແລຮັກສາ ແລະ​ ໃຊ້ Server ແຍກ​ອອກ​ໄປ​ຕ່າງ​ຫາກ​ອີກ​

    ຈາກ​ຈຸດ​ນີ້​ເອງທັງ​ການ​ເຮັດ Sharding ແລະ Caching ເປັນ​ສິ່ງ​ທີ່​ຖືກ​ພັດທະນາ​ຂຶ້ນ​ໃນ NoSQL ເທ​​ກ​ໂນ​ໂລ​​ຢີ ໂດຍ​ຮອງ​ຮັບ Auto-Sharding ແລະ Integrated Caching ໃນ​​ຕົວ​ເອງ ດັ່ງນັ້ນ​ເຮົາ​ຈຶ່ງ​ໄດ້​ເຫັນ NoSQL ຖືກ​ນຳໄປ​ໃຊ້​ງານ​ກັບ​ລະບົບ​ໃຫຍ່ໆເຊັ່ນ Facebook,Twitter, FourSquare, Digg ແລະ ​ອື່ນໆເພ​​າະວ່າ NoSQL ອອກ​ແບບ​ມາ​ເພື່ອ​ຮອງ​ຮັບ​ຄວາມ​ຕ້ອງການ​ງານ​ໃຫຍ່ໆ​ໄດ້​ດີ​ໂດຍ​ສະເພາະ​ຢູ່​ແລ້ວ ແຕ່​ເຖິງ​ຢ່າງ​ໃດກໍ​ຕາມ​ຍັງ​ມີ​ຄຸນສົມບັດ​ອື່ນໆທີ່​ໜ້າ​ສົນໃຈ​ໃນ NoSQL ເທັ​​ກ​ໂນ​ໂລ​​ຢີ

**ກ. ຄຸນສົມບັດ​ຂອງ NoSQL Database**

* **Dynamic Schemas**

    ການ​ຈັດ​ເກັບ​ຂໍ້​ມູນ​ຕ່າງ​ໆໃນ​ຖານ​ຂໍ້​ມູນ​ແບບ Relational Database ເຮົາ​ຈະ​ຕ້ອງ​ມີ​ການສ້າງ Schema ຫຼື ຮູບ​ແບບ​ຂອງ​ໂຄງ​ສ້າງ​ຕາ​ຕະລາງວ່າ​ຈະ​ຈັດ​ເກັບ​ຂໍ້​ມູນ​ຫັຍງ​ ເມື່ອ​ຕ້ອງ​ການ​ຈັດ​ເກັບ​ຂໍ້​ມູນ​ເພິ່ມເຕີມ​ຕ້ອງ​ປ່ຽນ Schema ພາຍ​ຫຼັງ (Alter-Table) ກ່ອນ​ຈະ​ຈັດ​ເກັບ​ຂໍ້​ມູນ​ຮູບ​ແບບ​ໃໝ່​ໄດ້.

    ແຕ່​ໃນ​ປະຈຸບັນການ​ຈັດ​ເກັບ​ຂໍ້​ມູນ​ມີ​ການ​ປ່ຽນ​ແປງ​ຕະຫລອດ​ເວລາ ເພາະ​ຄວາມ​ຕ້ອງ​ການ​ຈັດ​ເກັບ​ຂໍ້​ມູນ​ຕ່າງ​ໆມີ​ຫຼາກ​ຫຼາຍ​​ຂຶ້ນ​ເລື້ອຍ​ໆ ການ​ກຳນົດ​ໂຄງ​ສ້າງ​ຂອງ​ຕາ​ຕະລາງ​ຖານ​ຂໍ້​ມູນ ຫຼື ການ​ຕ້ອງ​ປ່ຽນ​ໂຄງ​ສ້າງ​ຖານ​ຂໍ້​ມູນ​ເລື້ອຍໆ ໂດຍ​ທີ່​ຂໍ້​ມູນ​ຍັງມີ​ຢູ່​ແລ້ວເປັນ​ເລື່ອງ​ທີ່​ຍາກ​ຫຼາຍ ​ຫຼື ເຮັດ​ບໍ່​ໄດ້​ເລີຍ ວິທີ​ການ​ຄື​ອາດຈະ​ຕ້ອງ​ແຍກ​ອອກ​ເປັນ​ຕາ​ຕະລາງ​ໃໝ່ຊຶ່ງ​ເປັນ​ວິທີ​ແກ້​ບັນຫາ​ຊົ່ວ​ຄາວ​ເທົ່າ​ນັ້ນ.

    ລະບົບ​ຖານ​ຂໍ້​ມູນ​ແບບ NoSQL ເຮົາ​ບໍ່​ຈຳ​ເປັນ​ຕ້ອງ​ມີ Schema ທີ່​ຕາຍ​ຕົວ ຫຼື ​ບໍ່​ຕ້ອງ​ມີ Schema ກ່ອນທີ່ຈະ​ຈັດ​ເກັບ​ຂໍ້​ມູນ ຂໍ້​ມູນ​ແຕ່​ລະ​ແຖວສາມາດ​ຈັດ​ເກັບ​ໄດ້​ຕາມ​ຕ້ອງ​ການ ຈະ​ເພີ່ມ​ ຫຼື ​ຫຼຸດ ກໍ​ບໍ່​ມີ​ບັນຫາ​ກັບ​ລະບົບ ເຮັດໃຫ້​ເຮົາ​ສາມາດ​ຈັດ​ເກັບ​ຂໍ້​ມູນ​ໄດ້​ຕາມ​ທີ່​ຕ້ອງ​ການປ່ຽນ​ແປງ​ໄດ້​ຕະຫລອດ​ເວລາ ສະ​ດວກ​ ແລະ ວ່ອງໄວ.

* **Auto-Sharding**

ເມື່ອ​ຂໍ້​ມູນ​ມີ​ຂະໜາດ​ໃຫຍ່ ຫຼື ​ເຮົາ​ຕ້ອງ​ການ​ເພີ່ມ​ປະ​ສິດ​ທິ​ພາບ​ການ ອ່ານ​ ແລະ ຂຽນ​ຂໍ້​ມູນ​ປະລິ ມານ​ຫຼາຍ​ໆ ການເຮັດ Sharding ໃນ​ລະບົບ NoSQL Database ຈະ​ກໍ່ການ​ກະ​ຈາຍ​ຂໍ້​ມູນ​ໄປ​ຍັງ Server ຕ່າງ​ໆ​ອັດ​ໂນ​ມັດ​​ (Auto-Sharding) ຜູ້​ພັດທະນາ (Developer) ບໍ່​ຕ້ອງ​ຂຽນ​ໂປຣ​ແກຣມ​ໃນ​ການ​ກະ​ຈາຍ​ຂໍ້​ມູນ​ເອງຄືກັບ Relational Database  
    ການ​ກະ​ຈາຍ​ຂໍ້​ມູນ​ອອກ​ໄປ​ຫຼາຍ​ໆ Server ນີ້​ຍັງ​ເຮັດໃຫ້​ມີ​ຂໍ້​ດີ​ຄື ປະ​ຢັດ​ຕົ້ນ​ທຶນ​ໃນ​ການ​ຂະຫຍາຍ​ລະບົບ ເພາະ​ເປັນ​ການ​ຂະຫຍາຍ​ແບບ​ແນວ​ນອນ (Scale Out) ຊຶ່ງ​ສາມາດ​ນຳ Server ປົກກະຕິ​ທົ່ວ​ໄປ​ມາ​ໃຊ້​ງານ​ໄດ້ ບໍ່​ຈຳ​ເປັນ​ຕ້ອງ​ເປັນ Enterprise Server

* **Replication**

ການ​ສຳ​ເນົາ​ຂໍ້​ມູນ​ຈາກ​ເຄື່ອງ​ໜຶ່ງ​ໄປ​ຢັງ​ອີກ​ເຄື່ອງ​ໜຶ່ງ (Replication) ເມື່ອ Server ໜຶ່ງ​ເສຍ​ຫາຍ ອີກ​ເຄື່ອງ​ໜຶ່ງ​ຈະ​ຂຶ້ນ​ມາ​ເຮັດວຽກ​ແທນ​ທັນ​ທີໂດຍ​ຂໍ້​ມູນ​ຂອງ​ແຕ່​ລະ​ເຄື່ອງ​ຈະ​ມີ​ຂໍ້​ມູນ​ຄືກັນ ດັ່ງ​ນັ້ນ Replication ເປັນ​ໜຶ່ງ​ຄຸນສົມບັດ​ທີ່​ຕອບສະໜອງ​ຕໍ່​ການ​ໃຊ້​ງານ​ທີ່​ຕ້ອງ​ການ​ຄວາມ​ຕໍ່​ເນື່ອງ​ໄດ້​ຕະຫລອດ​ເວລາ (High Availability)

* **Integrated Caching**

ການ​ຈັດ​ເກັບ​ຂໍ້​ມູນ​ທີ່​ໃຊ້​ງານ​ເລື້ອຍໆ ເຂົ້າ​ໄວ້​ໃນ Memory (RAM) ຊຶ່ງ​ເປັນ​ຄຸນສົມບັດ​ເດັ່ນ​ຂອງ NoSQL ທີ່​ທັງຫມົດ Caching ໄວ້​ໃນ​ຕົວ​ເອງ​ຢູ່​ແລ້ວ ເຮົາ​ບໍ່​ຈຳ​ເປັນ​ຕ້ອງ​ເຮັດ Cache Layer ຄືກັບ Relational Database ອີກ​ຕໍ່ໄປ ທີ່​ຕ້ອງ​ເຮັດ Cache Layer ແຍກ​ຕ່າງ​ຫາກ ​ແລະ​ ເບິ່ງແຍງຮັກສາ​ແຍກ​ອອກ​ໄປ​ຕ່າງ​ຫາກ​ອີກ​ດ້ວຍ

**ຂ. ປະ​ເພດ​ຂອງ​ຖານ​ຂໍ້​ມູນ NoSQL**

NoSQL ຖືກ​ແບ່ງ​ປະ​ເພດ​ຕາມ​ລັກສະນະ​ການ​ຈັດ​ເກັບ​ຂໍ້​ມູນ​ທີ່​ແຕກ​ຕ່າງ​ກັນ ດັ່ງ​ນັ້ນ​ການ​ຈະ​ເລືອກ NoSQL Database ໂຕໃດໂຕໜຶ່ງຈະ​ຕ້ອງ​ເບິ່ງ​ອີກ​ວ່າການ​ຈັດ​ເກັບ​ຂໍ້​ມູນ​ຂອງ​ຖານ​ຂໍ້​ມູນ​ເປັນ​ແບບໃດ ເຊັ່ນ

- Document databases ເຊັ່ນ MongoDB, CouchDB, Elasticsearch

- Graph stores ເຊັ່ນ Neo4J, Infinite Graph, InfoGrid

- Key-value stores ເຊັ່ນ DynamoDB, Redis, MemcacheDB

- Wide-column stores ເຊັ່ນ Cassandra, Amazon SimpleDB, Hadoop / HBase

**ຄ. Open source License**

    ໂດຍ​ສ່ວນ​ໃຫຍ່​ແລ້ວ NoSQL ຈະ​ເປັນ​ລິຂະ​ສິດ​ແບບ Open source ຊຶ່ງ​ບໍ່​ຕ້ອງ​ເສຍ​ຄ່າ​ໃຊ້​ຈ່າຍ​ໃນ​ການ​ນຳ​ມາ​ໃຊ້​ງານ ດັ່ງ​ນັ້ນ​ເຮົາ​ສາມາດ​ນຳ NoSQL Database ແຕ່​ລະ​ຕົວ​ມາ​ຕິດ​ຕັ້ງ​ໃຊ້​ງານ​ໄດ້​ໂດຍ​ບໍ່​ເສຍ​ຄ່າ​ໃຊ້​ຈ່າຍ​ໃດ​ໆ (ຟ​ຣີ)

**ງ. ນຳ NoSQL ມາ​ໃຊ້​ງານ​ຂະໜາດ​ນ້ອຍ​ໄດ້​ ຫຼື​ ບໍ່?**

    ຈາກ​ທີ່​ກ່າວ​ມາ​ແລ້ວ ຄົງ​ພໍ​ຈະ​ຕອບ​ຄຳ​ຖາມ​ນີ້​ໄດ້​ວ່າການ​ນຳ NoSQL Database ​ເມື່ອ​ນຳ​ມາ​ໃຊ້​ໃນ​ງານ​ຂະໜາດ​ໃຫຍ່​ນັ້ນເໝາະ​ສົມ​ຢ່າງ​ແນ່ນອນ ແຕ່​ຖ້າ​ເປັນ​ລະບົບ​ທົ່ວ​ໆ​ໄປ​ຄວນ​ຈະ​ນຳ NoSQL ມາ​ໃຊ້​ງານ ​ຫຼື ​ບໍ່

    ຄຳ​ຕອບ​ຄື ຂຶ້ນ​ຢູ່​ກັບ​ລັກສະນະ​ວຽກ​ວ່າເຮົາ​ຈະ​ໃຊ້​ຄຸນສົມບັດ​ຫັຍງ​ຂອງ NoSQL ຖ້າ​ເຮົາ​ຕ້ອງ​ການ​ຈັດ​ເກັບ​ຂໍ້​ມູນ​ທີ່​ບໍ່​ຕ້ອງ​ຢຶດ​ຕິດ​ກັບ​ໂຄງ​ສ້າງ (Dynamic Schema) ແລະ ​ຕ້ອງ​ການ​ເຂົ້າ​ໃຊ້​ງານ**​**ລະບົບ​ທີ່​ວ່ອງໄວ (Integrated Caching) ຂໍ້​ມູນ​ອາດຈະ​ຢັງ​ບໍ່​ຫຼາຍ​ເທົ່າ​ໃດອາດຈະ​ໃຊ້ NoSQL ໄດ້​ຢ່າງ​ແນ່ນອນ

ແຕ່​ຖ້າ​ບໍ່​ຕ້ອງ​ການ​ໃຊ້​ງານ (Dynamic Schema) ບໍ່​ຕ້ອງ​ການ​ເຂົ້າ​ໃຊ້​ງານ​ທີ່​ວ່ອງໄວ (Integrated Caching) ເພາະ​ໃຊ້ Relational Database ກໍ​ເຮັດ​ໄດ້​ດີ​ຢູ່​ແລ້ວ Database Server ກັບ Web Server ກໍ​ຢູ່​ທີ່ Server ດຽວ​ກັນ ຂໍ້​ມູນ​ບໍ່​ຫລາຍ​ນັ້ນ ບໍ່​ຕ້ອງ​ການ​ຈັດ​ເກັບ​ຂໍ້​ມູນ​ທີ່​ເພີ່ມ​ຂະຫຍາຍ​ຂຶ້ນ​ທຸກ​ມື້​ໆ ຜູ້​ເຂົ້າ​ໃຊ້​ງານ​ກໍ​ບໍ່​ໄດ້​ຫລາຍ ເບິ່ງ​ແລ້ວ​ວ່າ​ລະບົບ​​ບໍ່​ມີ​ແນວ​ໂນ້ມ​ຈະ​ຕ້ອງ​ຂະຫຍາຍ​ລະບົບ​ໃນ​ອະນາຄົດ​ອັນ​ໃກ້ ເຈົ້າ​ສາມາດ​ໃຊ້​ງານ Relational Database ໄດ້​ດີ​ຢູ່​ແລ້ວຢ່າງ​ບໍ່​ມີ​ບັນຫາ.

## ຄວາມຮູ້ພື້ນຖານກ່ຽວກັບຖານຂໍ້ມູນ MongoDB

## ຄວາມຮູ້ພື້ນຖານກ່ຽວກັບພາສາ JavaScript

## ຄວາມຮູ້ພື້ນຖານກ່ຽວກັບ Application Programming Interface (API)

## ຄວາມຮູ້ພື້ນຖານກ່ຽວກັບ Nodejs

Node.js ເປັນ open-source ແລະ cross-platform JavaScript runtime environment ທີ່​ກຳ​ລັງ​ໄດ້ຮັບ​ຄວາມ​ນິຍົມ​ສູງ ໂດຍ​ທົ່ວ​ໄປ​ເຮົາ​ຈະ​ໃຊ້ JavaScript ໃນ​ຝັ່ງ client ແຕ່ Node.js ເຮັດໃຫ້​ເຮົາ​ໃຊ້ JavaScript ໃນ​ຝັ່ງ Server ໄດ້​ດ້ວຍຊຶ່ງ Node.js ສາມມາດ run​ ໄດ້​ເທິງ platform ທີ່​ຫຼາກ​ຫຼາຍ​ທັງ Windows, Linux, Unix, Mac OS X ແລະ ອື່ນໆ.



ຮູບທີ 12 Nodejs

ຈຸດ​ເດັ່ນ​ທີ່ສຸດ​ຂອງ Node.js ຄື​ມັນ​ເຮັດວຽກ​ແບບ asynchronous ຜູ້​ອ່ານ​ອາດຈະ​ສົງ​ໃສ​ວ່າ​ແລ້ວ​ມັນ​ດີ​ແນວໃດ? ລອງ​ມາ​ເບິ່ງ​ຕົວ​ຢ່າງ​ການ​ຈັດການ​ໄຟ​ລ​ເມື່ອ​ມີ​ການ​ຮ້ອງ​ຂໍ​ຈາກ client ມາ​ທີ່ server ຂອງ PHP ກັບ Node.js ທຽບ​ກັນ​ເບິ່ງ​ເພື່ອ​ໃຫ້​ເຂົ້າໃຈ​ຫລາຍ​ຂຶ້ນ​

* ກໍລະນີ PHP ເມື່ອ​ມີ​ການ​ຮ້ອງ​ຂໍ​ເຂົ້າ​ມາ​ມັນ​ຈະ​ເຮັດ​ດັ່ງ​ນີ້:

1. ສົ່ງ task ໄປ​ທີ່​ລະບົບ​ໄຟ​ຂອງ​ຄອມພິວເຕີ
2. ລໍຖ້າຈົນກະທັ້ງ​ລະບົບ​ໄຟ​ລ​ເປີດ​ແລະ​ອ່ານ​ໄຟ​ລ​ສຳເລັດ
3. ສົ່ງ​ເນື້ອ​ຫາ​ຂອງ​ໄຟ​​ກັບ​ມາ​ໃຫ້ client
4. ພ້ອມ​ສຳລັບ​ຮັບ​ຄຳ​ຮ້ອງ​ຂໍ​ຖັດ​ໄປ

* ກໍລະນີ Node.js ເມື່ອ​ມີ​ການ​ຮ້ອງ​ຂໍ​ເຂົ້າ​ມາ​ມັນ​ຈະ​ເຮັດ​ດັ່ງ​ນີ້:

1. ສົ່ງ task ໄປ​ທີ່​ລະບົບ​ໄຟ​ຂອງ​ຄອມພິວເຕີ
2. ພ້ອມ​ສຳລັບ​ຮັບ​ຄຳ​ຮ້ອງ​ຂໍ​ຖັດ​ໄປ
3. ພໍ​ລະບົບ​ໄຟ​ລ​​ເປີດ​ແລະ​ອ່ານ​ໄຟ​ລ​ສຳເລັດ server ຈະ​ສົ່ງ​ເນື້ອ​ຫາ​ຂອງ​ໄຟ​ກັບ​ມາ​ໃຫ້ client

ຈາກ​ຕັວ​ຢ່າງ​ດ້ານ​ເທິງ​ຈະ​ເຫັນ​ວ່າ Node.js ຈະ​ຕັດ​ຂັ້ນ​ຕອນ​ການ​ລໍຖ້າຖີ້ມ​ແລ້ວ​ໄປ​ເຮັດ​ຄຳ​ຮ້ອງ​ຖັດ​ໄປ​ເລີຍທີ່ ​ເປັນ​ແບບນີ້​ເພາະ Node.js ຈະ​ run ​ແບບ single-threaded ແລະ​ ໃນ​ Libaryມາດຕະຖານ​ກໍ​ຈະ​ມີ​ເຊັດ​ຂອງ asynchronous I/O primitives ທີ່​ຊ່ວຍ​ປ້ອງ​ກັນ​ໂຄດ JavaScript ຈາກ​ການ blocking ເຮັດໃຫ້​ລະບົບ​ຄ່ອງແຄ້ວ ແລະ ​ມີ​ປະ​ສິດ​ທິ​ພາບ​ຫຼາຍ​ຂຶ້ນ

|  |  |
| --- | --- |
| **Synchronous vs Asynchronous** | |
| Synchronous ຄື​ການ​ run​ໂຄດ​ຕາມ​ລຳ​ດັບ​ທີ່​ເຮົາ​ຂຽນ​ໄວ້ ເຊັ່ນ  alert(1);  alert(2);  alert(3); ຜົນ​ລັບ​​ທີ່​ໄດ້​ຄື​ໜ້າ​ຈໍ​ຈະ​ສະແດງ​ຜົນ 1 ຈາກ​ນັ້ນ​ຈຶ່ງ​ສະແດງ​ຜົນ 2 ແລ້ວ​ສຸດ​ທ້າຍ​ຈຶ່ງ​ສະແດງ​ຜົນ 3 ຕາມ​ລຳ​ດັບ | Asynchronous ຄື​ການ​ຣັນໂຄ​ດ​ທີ່​ບໍ່​ຈຳ​ເປັນ​ຕ້ອງ​ເປັນ​ໄປ​ຕາມ​ລຳ​ດັບ​ທີ່​ເຮົາ​ຂຽນ​ໄວ້ ເຊັ່ນ:  alert(1);  setTimeout(() => alert(2), 0);  alert(3); ໃນ​ຕົວ​ຢ່າງ​ນີ້​ໂຄດ alert(2) ໃຊ້​ເວລາ​ດຳ​ເນີນ​ການ​ດົນ​ກວ່າ ເຮັດໃຫ້​ຜົນ​ລັບ​​ທີ່​ໄດ້​ຄື​ໜ້າ​ຈໍ​ຈະ​ສະແດງ​ຜົນ 1 ຈາກ​ນັ້ນ​ຈຶ່ງ​ສະແດງ​ຜົນ 3 ແລ້ວ​ສຸດ​ທ້າຍ​ຈຶ່ງ​ສະແດງ​ຜົນ 2 |
| **Blocking vs Non-blocking** | |
| Blocking ໝາຍ​ເຖິງ​ການ​ທີ່​ເຮົາ​ບໍ່​ສາມາດ​ດຳ​ເນີນ​ການ​ຕໍ່ໄປ​ໄດ້​ຈົນ​ກວ່າ​ຕົວ​ດຳ​ເນີນ​ການ​ທີ່​ກຳ​ລັງ​ລັນ​ຢູ່​ຈະ​ສຳເລັດ​ເສຍ​ກ່ອນ ເຊັ່ນ  alert(1);  var value = localStorage.getItem('foo');  alert(2); ຄຳ​ສັ່ງ localStorage ຈະ​ເປັນ​ຕົວ blocking ເຮັດໃຫ້​ບໍ່​ສາມາດ​ເຮັດ​ຄຳ​ສັ່ງ alert(2) ໄດ້​ຈົນ​ກວ່າ​ມັນ​ຈະ​ດຳ​ເນີນ​ການ​ສຳເລັດ ດັ່ງ​ນັ້ນ​ຜົນ​ລັບ​​ທີ່​ໄດ້​ຄື​ໜ້າ​ຈໍ​ຈະ​ສະແດງ​ຜົນ 1 ຈາກ​ນັ້ນ​ຈຶ່ງ​ລໍຖ້າຈົນກວ່າ​ຄຳ​ສັ່ງ localStorage ຈະ​ສຳເລັດ​ຈຶ່ງ​ສະແດງ​ຜົນ 2 | Non-blocking ໝາຍ​ເຖິງ​ການ​ໂຕທີ່​ດຳ​ເນີນ​ການ​ສາມາດ​ເຮັດ​ຄຳ​ສັ່ງ​ຖັດ​ໄປ​ໄດ້​ເລີຍ​ໂດຍ​ບໍ່​ຕ້ອງ​ລໍຖ້າ​ໃຫ້​ຄຳ​ສັ່ງ​ເດີມ​ເຮັດ​ສຳເລັດ​ກ່ອນເຊັ່ນ:  alert(1);  fetch('example.com').then(() => alert(2));  alert(3); ໃນ​ຕົວ​ຢ່າງ​ນີ້ ຄຳ​ສັ່ງ fetch ເປັນ non-blocking operation ດັ່ງ​ນັ້ນ​ຜົນ​ລັບ​ທີ່​ໄດ້​ຄື​ໜ້າ​ຈໍ​ຈະ​ສະແດງຜົນ 1 ຈາກ​ນັ້ນ​ຈຶ່ງ​ສະແດງ​ຜົນ 3 ແລ້ວ​ພໍ​ຄຳ​ສັ່ງ fetch ສຳເລັດ​ກໍ​ສະແດງ​ຜົນ 2 |

* **ປະຫວັດ Node.js**

ແຕ່​ເດີມ​ພາສາ JavaScript ຖືກ​ພັດທະນາ​ມາ​ເພື່ອ​ໃຊ້​ສຳລັບ Browser ທີ່​ຊື່ Netscape Navigator ໃນ​ປີ 1995 ​ໃນ​ຕອນ​ນັ້ນ Netscape ຕັ້ງ​ໃຈ​ຈະ​ຂາຍ Web Server ທີ່​ມີ environment ຊື່ Netscape LiveWire ຊຶ່ງ​ສາມາດ​ສ້າງ dynamic page ໂດຍ​ໃຊ້ JavaScript ທາງ​ຝັ່ງ server ອີກດ້ວຍ ແຕ່​ໜ້າ​ເສຍ​ດາຍ​ທີ່ Netscape LiveWire ບໍ່​ປະ​ສົບ​ຄວາມ​ສຳ​ເລັດ ແລະ​ການ​ໃຊ້ JavaScript ທາງ​ຝັ່ງ Server ກໍ​ບໍ່​ໄດ້ຮັບ​ຄວາມ​ນິຍົມ​ເລີຍ​ຈົນກະທັ້ງ Node.js ຖືກ​ກຳ​ເນີດ​ຂຶ້ນ​ມາ.

ສິ່ງ​ທີ່​ເຮັດໃຫ້ Node.js ເປັນ​ທີ່​ນິຍົມ​ຂຶ້ນ​ມາ​ຄື​ການ​ທີ່​ມັນ​ມາ​ໃນ​ຊ່ວງ​ເວລາ​ທີ່​ເໝາະ​ສົມ ເມື່ອ​ທຽບ​ກັບ JavaScript ທີ່​ເກີດ​ມາ​ຕັ້ງ​ແຕ່​ປີ 1995 ແລ້ວ Node.js ຫາກໍ່​ເກີດ​ມາ​ເມື່ອ​ປີ 2009 ເທົ່າ​ນັ້ນ ຕ້ອງ​ຂອບໃຈ "Web 2.0" applications ເຊັ່ນ Flickr, Gmail ແລະ ອື່ນໆທີ່​ສະແດງ​ໃຫ້​ໂລກ​ຮູ້​ວ່າ​ເວັບ​ສະ

ໄໝ​ໃໝ່​ຄວນ​ໜ້າ​ຕາ​ເປັນ​ຢ່າງໃດ

ບໍ່ດົນ​ກ່ອນ​ທີ່ Node.js ຈະ​ເກີດ ນັກ​ພັດທະນາ​ເບື້ອງ​ຫຼັງ browser ຊື່​ດັ່ງ​ທັງ​ຫຼາຍ​ແຂ່ງ​ຂັນ​ກັນ​ເຮັດວຽກ​ຢ່າງ​ໜັກ​ເພື່ອຈະ​ໃຊ້ JavaScript ໃຫ້​ໄດ້​ດີ​ທີ່ສຸດ ​ແລະ ​ຫາ​ທາງ​ເຮັດໃຫ້ JavaScript ສາມາດ​ລັນ​ໄດ້​ໄວ​ຫລາຍ​ຂຶ້ນເພື່ອ​ໃຫ້​ຜູ້​ໃຊ້​ງານ​ໄດ້ຮັບ​ປະ​ສິດ​ທິ​ພາບ​ທີ່​ດີ​ທີ່ສຸດ ຊຶ່ງ​ຜົນ​ຈາກ​ການ​ແຂ່ງ​ຂັນ​ນີ້​ເຮັດໃຫ້​ເກີດ​ການ​ພັດທະນາ Chrome V8 (open-source JavaScript engine ຂອງ The Chromium Project) ຂຶ້ນ​ມາ ແລະ Node.js ກໍ​ໃຊ້ engine ນີ້​​ເອງ

* ແຕ່​ການ​ທີ່ Node.js ເປັນ​ທີ່​ນິຍົມ​ຂຶ້ນ​ມາ​ບໍ່​ແມ່ນ​ແຕ່ວ່າ​ມັນ​ມາ​ຖືກ​ທີ່​ຖືກ​ເວລາ​ເທົ່າ​ນັ້ນ​ ແຕ່​ມັນ​ໄດ້​ສະແດງ​ໃຫ້​ເຫັນ​ແລ້ວ​ວ່າ ການ​ອອກ​ແບບ ​ແລະ ​ແນວ​ຄິດ​ຂອງ​ມັນ​ຊ່ວຍ​ນັກ​ພັດທະນາ​ທັງ​ຫຼາຍ​ໃຫ້​ສາມາດ​ໃຊ້ JavaScript ທາງ​ຝັ່ງ server ໄດ້​ງ່າຍ​ຂຶ້ນ​ຫລາຍ​ອີກດ້ວຍ​. (Nodejs ແມ່ນຫຍັງ [https://expert-programming-tutor.com/tutorial/article/A06\_NodeJS01\_HelloWorld.php 13 May 2017](https://expert-programming-tutor.com/tutorial/article/A06_NodeJS01_HelloWorld.php%2013%20May%202017)​).

## ຄວາມຮູ້ພື້ນຖານກ່ຽວກັບ Reactjs (JavaScript Library)

## ຄວາມຮູ້ພື້ນຖານກ່ຽວກັບ GraphQL (GraphQL API)

GraphQL ຄືພາສາ​ສຳລັບ​ການ​ເຂົ້າ​ເຖິງ​ຂໍ້​ມູນ (Query Language) ເພື່ອ​ການ​ໃຊ້​ງານ API ຂອງ​ລະບົບ ​ແລະ ​ຈະ​ປະ​ມວນ​ຜົນ​ຄຳ​ສັ່ງ​ທີ່​ຝັ່ງ server ຫຼື ​ທີ່​ເອີ້ນວ່າ server-side runtime ໂດຍ​ໃຊ້​ໂຄງ​ສ້າງ​ຂໍ້​ມູນ​ທີ່​ເຮົາ​ກຳນົດ​ໄວ້. ແຕ່ GraphQL ບໍ່​ໄດ້​ພັດທະນາ​ຂຶ້ນ​ມາ​ເພື່ອ​ແທນ​ທີ່​ພາສາ​ສຳລັບ​ການ​ເຂົ້າ​ເຖິງ​ຂໍ້​ມູນ​ເຊັ່ນ SQL ຫຼື ​ເຮັດ​ຫນ້າທີ່​ເປັນ​ລະບົບ​ຈັດ​ເກັບ​ຖານ​ຂໍ້​ມູນ (storage engine) ແຕ່​ຢ່າງ​ໃດ

ໃນ​ປະຈຸບັນ​ວິທີ​ທີ່​ເຂົ້າ​ເຖິງ​ຂໍ້​ມູນ​ຜ່ານ​ເວັບ​ເຊີ​ວິດ​ທີ່​ໄດ້ຮັບ​ຄວາມ​ນິຍົມ​ກໍ​ຈະ​ເປັນ REST API ຜ່ານ​ທາງ HTTP Mehtods ຕົວ​ຢ່າງ​ເຊັ່ນ ຕ້ອງ​ການ​ດຶງ​ຂໍ້​ມູນ​ຜູ້​ໃຊ້​ງານ​ທັງ​ໝົດ ກໍ​ສາມາດ​ຮຽກ​ຜ່ານ API ໄດ້​ດັ່ງ​ນີ້

GET /users

ຫຼື ຫາກ​ຕ້ອງ​ການ​ດຶງ​ເພື່ອນ​ທັງ​ໝົດ​ຂອງ​ຜູ້​ໃຊ້​ງານ ID 25

GET /users/25/friends

ແຕ່​ຫາກ​​ຕ້ອງ​ການ​ດຶງ​ເບີໂທ​ສັບ​ຂອງ​ເພື່ອນ​ທີ່​ເປັນ​ເພື່ອນ​ກັບ​ຜູ້​ໃຊ້​ງານ ID 25 ຈະ​ຕ້ອງ​ດຶງ ​ແລະ ​ຂຽນ​ໂປຣ​ແກຣມ​ແບບໃດ? ໂດຍ​ຜົມ​ມີ​ການ​ຈັດ​ເກັບ​ຂໍ້​ມູນ​ດັ່ງ​ນີ້

ຈາກ​ຕົວ​ຢ່າງ​ນີ້​ຈະ​ເຫັນ​ໄດ້​ວ່າ​ເຮົາ​ຈຳ​ເປັນ​ຕ້ອງ​ມີ API 2 end point ນຳກັນ ຄື GET /users/:id/friends ເພື່ອ​ດຶງ​ຂໍ້​ມູນ​ເພື່ອນ​ຂອງ​ຜູ້​ໃຊ້ ID 25 ຈາກ​ນັ້ນ​ກໍ​ວົນ​ລູບ​ດຶງ​ຂໍ້​ມູນ​ຜູ້​ໃຊ້​ໂດຍ​ໃຊ້ GET /users/:id ເພື່ອ​ດຶງ​ຂໍ້​ມູນ​ເບີໂທສັບ​ອີກຄັ້ງ ແລະ ​ວິທີ​ນີ້​​ກໍ​ຈະ​ໄດ້​ຂໍ້​ມູນ​ອື່ນ​ໆ ​ທີ່​ບໍ່​ຈຳ​ເປັນ​ຂອງ​ຜູ້​ໃຊ້​ມາ​ອີກ​ດ້ວຍ ບໍ່​ວ່າ​ຈະ​ເປັນ​ຊື່, ຮູບ​ພາບ ແລະ ​ອີ​ເມ​ວ

ໃນ​ຄວາມ​ເປັນ​ຈິງ​ແລ້ວ ອົງ​ກອນໃຫຍ່​ໆທີ່​ມີ​ຫຼາກ​ຫຼາຍ​ພະແນກ​ ຕ່າງ​ກໍ​ຮ້ອງ​ຂໍ​ຂໍ້​ມູນ​ທີ່​ບໍ່​ຄືກັນ ເຊັ່ນ ບາງ​ພະແນກຕ້ອງ​ການ​ສະເພາະ​ຂໍ້​ມູ​ນພະ​ນັກ​ງານ ບາງ​ພະແນກຕ້ອງ​ການ​ຂໍ້​ມູ​ນພະນັກ​ງານ​ພ້ອມ​ຂໍ້​ມູນ​ເງິນ​ເດືອນ ຫຼື ບາງ​ພະແນກ​ຕ້ອງ​ການ​ຂໍ້​ມູ​ນພະ​ນັກ​ງານ​ກັບ​ຈຳນວນ​ວັນທີ​ຂາດ​ ຫຼື ​ລາ​ເທົ່າ​ນັ້ນເປັນ​ຕົ້ນ ການ​ໃຊ້​ງານ REST API ຈຶ່ງ​ສ້າງ​ຄວາມ​ລຳ​ບາກ​ຕໍ່​ການ​ພັດທະນາ ​ແລະ ​ດູແລຮັກສາ​ເຊີ​ວິດ.

ດ້ວຍ​ເຫດ​ນີ້ GraphQL ຈຶ່ງ​ໄດ້​ເຂົ້າ​ມາ​ເພື່ອ​ແກ້​ບັນຫາ​ໃນ​ຈຸດ​ນີ້​ໂດຍ​ສະເພາະ ເພື່ອ​ໃຫ້​ເຮົາ​ສາ ມາດ​ດຶງ​ຂໍ້​ມູນ​ໄດ້​ກົງ​ກັບ​ຄວາມ​ຕ້ອງ​ການ ຫຼຸດ​ຄວາມ​ຊັບ​ຊ້ອນ​ໃນ​ການ​ຂຽນ​ໂຄດ​ເພື່ອ​ດຶງ​ຂໍ້​ມູນ ສາມາດ​ຈັດການ​ ແລະ ​ດູແລຮັກສາ​ໂຄ້​ດ​ໃນ​ຝັ່ງ server-side ໄດ້​ງ່າຍ​ຫລາຍ​ຢິ່ງ​ຂຶ້ນ ແລະ ​ທີ່​ສຳຄັນ GraphQL ບໍ່​ໄດ້​ຜູກ​ຕິດ​ກັບ database ແລະ ທີ່​ຈັດ​ເກັບ​ຂໍ້​ມູນ​ໃດ​ໆ​ທັງ​ນັ້ນ

**(ທຳຄວາມຮູ້ຈັກກັບ GraphQL** <https://medium.com/@athivvat/ทำความรู้จักกับ-graphql-980ac00d6c17#:~:text=GraphQL%20คือ%20ภาษาสำหรับการเข้าถึงข้อมูล%20(Query%20Language)%20เพื่อ,ข้อมูลที่เรากำหนดไว้> **23 Feb 2020​).**

## ຄວາມຮູ້ພື້ນຖານກ່ຽວກັບ expressjs (Nodejs Library)

## ຄວາມຮູ້ພື້ນຖານກ່ຽວກັບ Progressive Web App (PWA)

## ຄວາມຮູ້ກ່ຽວກັບໂປຣແກຣມ Microsoft Office 2016

## ຄວາມຮູ້ກ່ຽວກັບໂປຣແກຣມ Microsoft Visio 2016

## ຄວາມຮູ້ກ່ຽວກັບໂປຣແກຣມ Adobe XD CC

## ຄວາມຮູ້ກ່ຽວກັບໂປຣແກຣມ Visual Studio Code

# **ບົດທີ 3**

# **ວິທີດຳເນີນການຄົ້ນຄວ້າ**

## ວິທີສຶກສາ ແລະ ຄົ້ນຄວ້າ

### ໄລຍະທີ່ 1

### ໄລຍະທີ່ 2

### ໄລຍະທີ່ 3

### ໄລຍະທີ່ 4

### ໄລຍະທີ່ 5

* 1. **ເຄື່ອງມືທີ່ໃຊ້ໃນການພັດທະນາ (Development Tools)**

1. **Hardware:**

* ເຄື່ອງຄອມພີວເຕີ 1 ໜ່ວຍລຸ້ນ Lenovo Lenovo Intel(R) Core(TM) I5-3230M
* ເຄື່ອງຄອມພີວເຕີທີ່ໃຊ້ສຳລັບຂຽນໂປຣແກຣມມີ Specດັ່ງນີ້:
* CPU @2.60GHz,
* RAM 8GB DDR3L 1600MHz.
* SSD 240GB,HDD 500GB
* Pocket WiFi Modem ເພື່ອໃຊ້ເຊື່ອມຕໍ່ Internet.

1. **Software:**

* ລະບົບປະຕິບັດການ Windows 10 Professional 64 Bit
* Microsoft Visio 2016 ໃຊ້ແຕ້ມແຜນວາດການໄຫຼຂໍ້ມູນ (DFD, ER, Flowchart)
* Adobe XD ໃຊ້ອອກແບບ UX/UI
* Studio 3T For MongoDB ແລະ Moon Modeler ໃຊ້ຈັດການຖານຂໍ້ມູນ ແລະ ອອກແບບ Database Model
* Visual Studio Code ໃຊ້ຂຽນໂຄດດ້ວຍພາສາ JavaScript (ReactJS, NodeJS, GraphQL)
* MS Office 2013 Professional ໃຊ້ເພື່ອສ້າງ
  1. **ການວິເຄາະຂໍ້ມູນ**
     1. **ແຜນວາດລວມຂອງລະບົບ**
     2. **ຕາຕະລາງສະແດງລາຍລະອຽດຕ່າງໆທີ່ກ່ຽວກັບລະບົບ**
     3. **ແຜນວາດເນື້ອຫາ (Context Diagram)**
     4. **ແຜນວາດລຳດັບຊັ້ນໜ້າທີ່ (Functional Hierarchy Diagram)**
     5. **ແຜນວາດການໄຫຼຂໍ້ມູນ (Data Flow Diagram: DFD)**
  2. **ການອອກແບບລະບົບ**
     1. **ການອອກແບບຮ່າງສະແດງຜົນ (Output Design)**
     2. **ການອອກແບບຮ່າງປ້ອນຂໍ້ມູນ (Input Design)**
     3. **ການອອກແບບຖານຂໍ້ມູນ**
  3. **f**

# 

# **ບົດທີ 4**

# **ຜົນຂອງການວິເຄາະຂໍ້ມູນ ແລະ ການອະທິບາຍຜົນ**

### 4.1 ໜ້າຕ່າງການ (Login Form)

ເມື່ອເປີດໂປຣແປຣມຂຶ້ນມາກໍ່ຈະເຫັນວ່າເຂົ້າສູ່ລະບົບເປັນໜ້າທຳອິດຈາກນັ້ນຈຶ່ງທຳການເຂົ້າສູ່ລະບົບດ້ວຍຊື່ ແລະລະຫັດເຂົ້າໃຊ້ເພື່ອເຂົ້າໄປດ້ານໃນຂອງໂປຣແກຣມດັ່ງຮູບລຸ່ມນີ້:

- ວິທີການເຂົ້າສູ່ລະບົບມີຄື :

1) ໃສ່ຊື່ຜູ້ໃຊ້ລະບົບ.

2) ໃສ່ລະຫັດຜ່ານ.

3) ກົດປຸ່ມເພື່ອເຂົ້າລະບົບ.

4) ຖ້າຊື່ກັບລະຫັດຖືກຈະເປີດນຳໃຊ້ໂປຣແກຣມໄດ້ປົກກະຕິແລ້ວ.

5) ຖ້າຊື່ກັບລະຫັດຜິດຈະມີຂໍ້ຄວາມຂື້ນມາວ່າ ກະລຸນາກວດສອບຂໍ້ມູນຂອງທ່ານໃຫ້ຖືກຕ້ອງ ແລ້ວກົດປຸ່ມ OK

6) ຖ້າບໍ່ເຂົ້າລະບົບແມ່ນກົດປຸ່ມອອກລະບົບ.

ຮູບທີ 4.1 ຟອມຫຼັກ (Login)

### 4.2 ຟ້ອນຫຼັກ (Main Form)

ຟອມຫຼັກແມ່ນຟອມທີ່ເຮົາສາມາດເຂົ້າຫາທຸກໆຟອມໄດ້ເຊັ່ນ ຟອມຈັດການຂໍ້ມູນ, ຟອມປ້ອນຂໍ້ມູນ, ຟອມການບໍລິການ, ຟອມລາຍຮັບ, ຟອມລາຍຈ່າຍ, ຟອມລາຍງານ.

ຮູບທີ 4.2 ຟອມຫຼັກ (Main Form)

### 4.3 ໜ້າຟອມຈັດການຫ້ອງພັກຂອງໂປຣມແກຣມ

ເມື່ອເຮົາເຂົ້າມາໃນໜ້າຫຼັກຂອງໂປຣແກຣມແລ້ວຖ້າເລືອກການຈັດການຫ້ອງພັກເຮົາກໍ່ຈະເຫັນດັ່ງຮູບລຸ່ມນີ້: ເຊັ່ນວ່າເຮົາປ້ອນຂໍ້ມູນຫ້ອງ, ລະຫັດຫ້ອງ, ຊື່ເບີຫ້ອງ, ສະຖານະ, ຂໍ້ມູນປະເພດຫ້ອງ.

ຮູບທີ 4.3 ຮູບໜ້າຟອມຫ້ອງພັກ

### 4.4 ໜ້າຟອມຈັດການປະເພດຫ້ອງຂອງໂປຣມແກຣມ

ເມື່ອເຮົາເຂົ້າມາໃນໜ້າຫຼັກຂອງໂປຣແກຣມແລ້ວຖ້າເລືອກການຈັດການປະເພດຫ້ອງພັກເຮົາກໍ່ຈະເຫັນດັ່ງຮູບລຸ່ມນີ້:

ຮູບທີ 4.4 ຮູບໜ້າຟອມປະເພດຫ້ອງພັກ

### 4.5 ໜ້າຟອມຈັດການລູກຄ້າຂອງໂປຣມແກຣມ

ເມື່ອເຮົາເຂົ້າມາໃນໜ້າຫຼັກຂອງໂປຣແກຣມແລ້ວຖ້າເລືອກການຈັດການຂໍ້ມູນລູກຄ້າເຮົາກໍ່ສາມາດຄົ້ນຫາຕາມລະຫັດ ຫຼື ຄົ້ນຫາຕາມຊື້.

ຮູບທີ 4.5 ຮູບໜ້າຟອມຈັດການຂໍ້ມູນລູກຄ້າ

### 4.6 ໜ້າຟອມການຈັດການຂໍ້ມູນພະນັກງານ

ເມື່ອເຮົາເຂົ້າມາໃນໜ້າຫຼັກຂອງໂປຣແກຣມແລ້ວຖ້າເລືອກການຈັດການຂໍ້ມູນພະນັກງານເຮົາກໍ່ສາມາດປ້ອນຂໍ້ມູນພະນັກງານບໍ່ວ່າຈະຄົ້ນຫາດ້ວຍການໃສ່ລະຫັດ ຫຼື ຄົ້ນຫາຈາກຊື້, ການເພີ່ມ, ແກ້ໄຂ, ບັນທືກ ແລະ ລຶບກໍ່ໄດ້.

ຮູບທີ 4.6 ຮູບໜ້າຟອມຈັດການຂໍ້ມູນພະນັກງານ

### 4.7 ໜ້າຟອມການບໍລິການ

ເມື່ອເຮົາເຂົ້າມາໃນໜ້າຫຼັກຂອງໂປຣແກຣມແລ້ວຖ້າເລືອກການຈັດການບໍລິການເຮົາກໍ່ຈະເຫັນລາຍລະອຽດດັ່ງຮູບລຸ່ມນີ້:

ຮູບທີ 4.7 ຮູບໜ້າຟອມການບໍລິການ

### 4.8 ໜ້າຟອມການຈັດການ ການຈອງ

ເມື່ອເຮົາເຂົ້າມາໃນໜ້າການຈອງຂອງໂປຣແກຣມແລ້ວຖ້າເລືອກການຈັດການຈອງເຮົາກໍ່ຈະເຫັນລາຍລະອຽດດັ່ງຮູບລຸ່ມນີ້

ຮູບທີ 4.8 ຮູບໜ້າຟອມການຈອງ

### 4.9 ໜ້າຟອມການແຈ້ງອອກ

ເມື່ອເຮົາເຂົ້າມາໃນໜ້າຫຼັກຂອງໂປຣແກຣມແລ້ວຖ້າເລືອກຫນ້າຟອມການແຈ້ງອອກເຮົາກໍ່ຈະເຫັນລາຍລະອຽດດັ່ງຮູບລຸ່ມນີ້:

ຮູບທີ 4.9 ຮູບໜ້າຟອມການແຈ້ງອອກ

### 4.10 ໜ້າຟອມຄົ້ນຫາລູກຄ້າ

ເມື່ອເຮົາເຂົ້າມາໃນໜ້າຫຼັກຂອງໂປຣແກຣມແລ້ວຖ້າເລືອກຫນ້າຟອມຄົ້ນຫາລູກຄ້າເຮົາກໍ່ຈະເຫັນລາຍລະອຽດດັ່ງຮູບລຸ່ມນີ້:

ຮູບທີ 4.10 ຮູບໜ້າຟອມຄົ້ນຫາລູກຄ້າ

### 4.11 ໜ້າຟອມຄົ້ນຫາພະນັກງານ

ເມື່ອເຮົາເຂົ້າມາໃນໜ້າຫຼັກຂອງໂປຣແກຣມແລ້ວຖ້າເລືອກຫນ້າຟອມຄົ້ນຫາພະນັກງານເຮົາກໍ່ຈະເຫັນລາຍລະອຽດດັ່ງຮູບລຸ່ມນີ້:

ຮູບທີ 3.11 ຮູບໜ້າຟອມຄົ້ນຫາພະນັກງານ

### 4.12 ໜ້າຟອມຄົ້ນຫາ ຂໍ້ມູນຫ້ອງ

ເມື່ອເຮົາເຂົ້າມາໃນໜ້າຫຼັກຂອງໂປຣແກຣມແລ້ວຖ້າເລືອກຫນ້າຟອມຄົ້ນຫາຂໍ້ມູນຫ້ອງພັກເຮົາກໍ່ຈະເຫັນລາຍລະອຽດດັ່ງຮູບລຸ່ມນີ້:

ຮູບທີ 4.12 ຮູບໜ້າຟອມຄົ້ນຫາຂໍ້ມູນຫ້ອງພັກ

### 4.13 ໜ້າຟອມຄົ້ນຫາ ການແຈ້ງອອກ

ເມື່ອເຮົາເຂົ້າມາໃນໜ້າຫຼັກຂອງໂປຣແກຣມແລ້ວຖ້າເລືອກຫນ້າຟອມຄົ້ນຫາຂໍ້ມູນການແຈ້ງອອກເຮົາກໍ່ຈະເຫັນລາຍລະອຽດດັ່ງຮູບລຸ່ມນີ້:

ຮູບທີ 4.13 ຮູບໜ້າຟອມຄົ້ນຫາຂໍ້ມູນການແຈ້ງອອກ

### 4.14 ໜ້າຟອມການລາຍງານ

ເມື່ອເຮົາເລືອກໄປທີ່ການລາຍງານ ເຮົາກໍ່ຈະເຫັນລາຍລະອຽດຂອງການລາຍງານກໍ່ຈະມີການຕັ້ງຄ່າ, ຈັດການຂໍ້ມູນ, ການບໍລິການ, ຄົ້ນຫາ, ພິມລາຍງານ.

ຮູບທີ 4.14 ຮູບໜ້າຟອມການລາຍງານ

1. **ການລາຍງານລູກຄ້າ**

ເມື່ອເຮົາເລືອກໄປທີ່ການລາຍງານລູກຄ້າເຮົາກໍ່ຈະເຫັນລາຍລະອຽດຂອງການລາຍງານດັ່ງຮູບລຸ່ມນີ້:

ຮູບທີ 4.15 ຮູບໜ້າຟອມການລາຍງານລູກຄ້າ

* **ການລາຍງານການຈອງ**

ເມື່ອເຮົາເລືອກໄປທີ່ການລາຍງານການຈອງ ເຮົາກໍ່ຈະເຫັນລາຍລະອຽດຂອງການລາຍງານດັ່ງຮູບລຸ່ມນີ້:

ຮູບທີ 4.16 ຮູບໜ້າຟອມການລາຍງານການຈອງ

1. **ການລາຍງານການເຂົ້າພັກ**

ເມື່ອເຮົາເລືອກໄປທີ່ການລາຍງານເຂົ້າພັກເຮົາກໍ່ຈະເຫັນລາຍລະອຽດຂອງການລາຍງານດັ່ງຮູບລຸ່ມນີ້:

ຮູບທີ 4.17 ຮູບໜ້າຟອມການລາຍງານການຈອງ

1. **ການລາຍງານການແຈ້ງອອກ**

ເມື່ອເຮົາເລືອກໄປທີ່ການລາຍງານການແຈ້ງອອກເຮົາກໍ່ຈະເຫັນລາຍລະອຽດຂອງການລາຍງານດັ່ງຮູບລຸ່ມນີ້:

ຮູບທີ 4.18 ຮູບໜ້າຟອມການລາຍງານການແຈ້ງອອກ

1. **ການລາຍງານລາຍຮັບວັນ/ເດືອນ/ປີ**

ເມື່ອເຮົາເລືອກໄປທີ່ການລາຍງານລາຍຮັບວັນ/ເດືອນ/ປີເຮົາກໍ່ຈະເຫັນລາຍລະອຽດຂອງການລາຍງານດັ່ງຮູບລຸ່ມນີ້:

ຮູບທີ 4.19 ຮູບໜ້າຟອມການລາຍງານລາຍຮັບວັນ/ເດືອນ/ປີ

1. **ການລາຍງານຫ້ອງພັກ**

ເມື່ອເຮົາເລືອກໄປທີ່ການລາຍງານຫ້ອງພັກເຮົາກໍ່ຈະເຫັນລາຍລະອຽດຂອງການລາຍງານດັ່ງຮູບລຸ່ມນີ້:

ຮູບທີ 4.20 ຮູບໜ້າຟອມການລາຍງານຫ້ອງພັກ

1. **ການລາຍງານປະເພດຫ້ອງພັກ**

ເມື່ອເຮົາເລືອກໄປທີ່ການລາຍງານປະເພດຫ້ອງພັກເຮົາກໍ່ຈະເຫັນລາຍລະອຽດຂອງການລາຍງານດັ່ງຮູບລຸ່ມນີ້:

ຮູບທີ 4.21 ຮູບໜ້າຟອມການລາຍງານຫ້ອງພັກ

1. **ໃບບິນ**

ຮູບທີ 4. 22 ຮູບໃບບິນ

# **ບົດທີ 5**

# **ສະຫຼຸບ ແລະ ຂໍ້ສະເໜີ**

## 5.1 ສະຫຼຸບ

ລະບົບຂາຍປີ້ລົດເມອອນໄລຂອງສະຖານີຂົນສົ່ງໂດຍສານສາຍໃຕ້ແມ່ນຖືກພົດທະນາຂຶ້ນເພື່ອຊ່ວຍ ໃຫ້ສະຖານີມີລະບົບທີ່ທັນສະໄໝສະດວກສະບາຍ ແລະ ເຮັດ ໃຫ້ການຈັດການຂໍ້ມູນ, ການບໍລິການພາຍໃນ ສະຖານີຢ່າງວ່ອງໄວ, ມີຄວາມເປັນລະບຽບ, ສະດວກໃນການລາຍງານ, ຂໍ້ມູນທີ່ມີຄວາມຖືກຕ້ອງ ແລະ ຊັດເຈນ. ຂອບເຂດຂອງການດໍາເນີນວຽກງານໂປຣແກຣມ ຂອງພວກຂ້າພະເຈົ້າມີຈັດການຂໍ້ມູນ , ສະໝັກສະມາຊິກ , ບໍລິການ ແລະ ລາຍງານ. ເຊິ່ງເປັນການທົດແທນການເຮັດວຽກໃນລະບົບເກົ່າໂດຍລະບົບໃໝ່ເພື່ອຫຼຸດຜ່ອນບັນຫາການເສຍຫາຍຂອງຂໍ້ມູນ. ພວກນ້ອງສາມາດພັດທະນາໂປຣແກຮມ ເພື່ອຊ່ວຍໃຫ້ເຮັດວຽກສະດວກສະບາຍຍິ່ງຂື້ນ ແລະ ໄດ້ອອກແບບໜ້າຟອມການປ້ອນຂໍ້ມູນຕ່າງໆ.

- ຟອມຈັດການຂໍ້ມູນຫຼັກ 3 ຟອມ

- ສ້າງຟອມຈັດການໄດ້ 5 ຟອມ

- ຟອມການຄົ້ນຫາໄດ້ 5 ຟອມ

- ພິມລາຍງານທັງໝົດໄດ້ 9 ລາຍງານ

## 5.2 ຈຸດດີ

- ຮູ້ວິເຄາະຫາບັນຫາ ແລະ ສາເຫດຂອງລະບົບເກົ່າ

- ຈະໄດ້ລະບົບຈອງປີ້ລົດເມສາຍໃຕ້ແບບອອນໄລໃໝ່

- ສາມາດນໍາໃຊ້ເວບໄຊທີ່ສ້າງຂຶ້ນມາເຂົ້າຊ່ວຍໃນການຈອງປີ້ລົດເມແບບອອນລາຍ

- ເວບໄຊທີ່ສ້າງຂຶ້ນມາສາມາດຈອງອອນລາຍໄດ້

- ການເຮັດບົດລາຍງານສະດວກສະບາຍ ແລະ ວ່ອງໄວຂື້ນ

## 5.3 ຈຸດອ່ອນ

ໂປແກມນີ້ຍັງບໍ່ສົມບູນ ແລະ ຄົບຖ້ວນຕາມຄວາມຕ້ອງການຂອງສະຖານີຂົນສົ່ງໂດຍສານສາຍໃຕ້ເທື່ອ ເນື່ອງຈາກວ່າໃນການສຶກສາຍັງບໍ່ທັນມີປະສົບການໃນການຂຽນເວບໄຊມາກ່ອນ,ຍັງບໍ່ຄວບຄຸມເຖິງຄວາມຕ້ອງການຕົວຈິງຂອງຜູ້ໃຊ້ລະບົບ.

## 5.4 ແນວທາງໃນການພັດທະນາ ແລະ ຂະຫຍາຍຕໍ່ຂອງສະຖານີ

ເນື່ອງຈາກວ່າໂປແກມນີ້ເປັນໂປຣແກຣມທີ່ສ້າງຂື້ນມາແລ້ວ ດັ່ງນັ້ນເພື່ອເປັນການກວດສວບຫາຂໍ້ພິດພາດ ແລະ ເພື່ອຫາຈຸດດີຈຸດອ່ອນມາທຳການປັບປຸງ ແກ້ໄຂຈື່ງຄວນນຳເອົາໂປຣແກຣມນີ້ໄປປັບປຸງຈຸດບົກພ່ອງຂອງໂປແກຣມໃນບາງສ່ວນ ເພື່ອເຮັດໃຫ້ໂປຣແກຣມມີປະສິດທິພາບ ແລະ ເຮັດວຽກໄດ້ດີຂື້ນ ເພື່ອຈະນຳເອົາໄປປະກອບສ່ວນເຂົ້າໃນການເຮັດວຽກງານຕົວຈິງ

ດັ່ງນັ້ນໂປຣແກຣມນີ້ເປັນໂປຣແກຣມໜື່ງທີ່ມີຄຸນສົມບັດໃນການໃຊ້ຖານຂໍ້ມູນໄດ້.

**ເອກະສານອ້າງອີງ**

**ປະຫວັດຫຍໍ້ຜູ້ຂຽນ**

ຊື່ ແລະ ນາມສະກຸນ: ທ້າວ ມະໂນພອນ ມະໂນກຸນ

ວັນ,ເດືອນ, ປີເກີດ: 16 ເດືອນ ທັນວາ ປີ 1995

ບ້ານເກິດ: ທົ່ງຂັນຄຳ ເມືອງ: ຈັນທະບູລີ ແຂວງ: ນະຄອນ

ຫຼວງວຽງຈັນ ບ້ານຢູ່ປັດຈຸບັນ: ທົ່ງຂັນຄຳ ເມືອງ: ຈັນທະບູລີແຂວງ: ນະຄອນຫຼວງວຽງຈັນ

ການສຶກສາ: ປີ 2016 ຈົບຊັ້ນສູງ ທີ່ ສະຖາບັນພັດທະນາສີມືແຮງງານລາວ - ເກົາຫຼີ

ປີ 2013 ຈົບມັດທະຍົມຕອນປາຍ ມ.ສ ເຈົ້າອານຸວົງ

ປີ 2010 ຈົບມັດທະຍົມຕອນຕົ້ນ ທີ ມ.ສ ເຈົ້າອານຸວົງ

ປີ 2007 ຈົບປະຖົມສົມບູນ ທີ ໂຮງຮຽນປະຖົມສົມບູນອານຸ

ເບີໂທ: 020 5400 0003 , 020 2814 9278

ອີເມວ: [dou](mailto:lakitptt@gmail.com)\_2020@hotmail.com

**ປະຫວັດຫຍໍ້ຜູ້ຂຽນ**

ຊື່ ແລະ ນາມສະກຸນ: ທ້າວ ພອນຄຳ ແກ້ວມະນີ

ວັນ,ເດືອນ,ປີເກີດ: 07 ເດືອນ ມີນາ ປີ 1995

ບ້ານເກິດ:ບ້ານ ທ້ຽວ, ເມືອງ: ວຽງພູຄາ, ແຂວງ: ຫຼວງນໍ້າທາ ບ້ານຢູ່ປັດຈຸບັນ:ບ້ານ ທ້ຽວ, ເມືອງ: ວຽງພູຄາ, ແຂວງ: ຫຼວງນໍ້າທາ

ການສຶກສາ: ປີ 2016 ຈົບຊັ້ນສູງ ທີ່ ສະຖາບັນພັດທະນາສີມືແຮງງານລາວ - ເກົາຫຼີ

ປີ 2013 ຈົບມັດທະຍົມສົມບູນເມືອງວຽງພູຄາ

ປີ 2007 ຈົບປະຖົມສົມບູນ ທີ ໂຮງຮຽນປະຖົມສົມບູນບ້ານດົງວຽງ

ເບີໂທ: 020 2802 2677

ອີເມວ: [phonekham.dev](mailto:lakitptt@gmail.com)@gmail.com