

## ບົດທີ 1

### ບົດສະເໜີ

#### 1.1 ຄວາມສໍາຄັນຂອງບັນຫາ

ອີງໃສ່ເພື່ອຄວາມສະດວກສະບາຍຂອງປະຊາຊົນລາວເຮົາ ແລະ ຄົນຕ່າງປະເທດ, ນັກທ່ອງທ່ຽວ, ພະນັກງານ, ພໍ່ຄ້າຊາວຂາຍຕະຫຼອດຮອດພໍ່ແມ່ປະຊາຊົນທີ່ເດີນທາງເຂົ້າມາທ່ອງທ່ຽວ ຫຼື ມາເຮັດວຽກ ເຮັດງານທຳນະຄອນຫຼວງວຽງຈັນຂອງພວກເຮົາເພື່ອໃຫ້ການໄປມາສະຖານທີ່ຕ່າງໆພວກເຮົາຈຶ່ງໄດ້ ສ້າງຕັ້ງສະຖານນີຂົນສົ່ງໂດຍສານນີ້ຂຶ້ນມາ.

ສະຖານນີຂົນສົ່ງໂດຍສານສາຍໃຕ້ ເປັນບ່ອນບໍລິການຮັບ-ສົ່ງຜູ້ໂດຍສານ, ສິນຄ້າວັດຖຸສິ່ງຂອງ ແລະ ສັດ ຈາກຈຸດໜຶ່ງໄປຫາອີກຈຸດໜຶ່ງ ຊຶ່ງສະຖານນີຂົນສົ່ງໂດຍສານທາງໄກສາຍໃຕ້ນີ້ແມ່ນໄດ້ສ້າງ ຕັ້ງຂຶ້ນໃນວັນທີ 1 ກັນຍາ 2016 ເຊິ່ງມີລາຍລະອຽດດັ່ງລຸ່ມນີ້: ທີ່ຕັ້ງ ແລະ ພາລະບົດບາດ ຂອງສະ ຖານນີຂົນສົ່ງທາງໄກສາຍໃຕ້ແມ່ນສະຖານນີໜຶ່ງຊຶ່ງຕັ້ງຢູ່ ບ້ານ ສະພັງມຶກ, ເມືອງ ໄຊທານີ, ນະຄອນ ຫຼວງວຽງຈັນ, ຖະໜົນເລກທີ 450 ປີ ໃກ້ກັບ ສີ່ແຍກໄຟແດງດົງໂດກ. ສະຖານນີຂົນສົ່ງທາງໄກສາຍ ໃຕ້ ປະກອບມີຫຼາຍໜ່ວຍງານຄື: ອຳນວຍການໃຫຍ່ມີ 1 ທ່ານ, ເລຂານຸການມີ 1 ທ່ານ, ໜ່ວຍງານ ແຜນການມີ 1 ທ່ານ, ໜ່ວຍງານຮັບ-ຈ່າຍເງິນມີ 1 ທ່ານ, ໜ່ວຍງານບໍລິການຂາຍປີ້ມີ 7 ທ່ານ, ໜ່ວຍ ງານຮັກສາຄວາມປອດໄພມີ 6 ທ່ານ, ໜ່ວຍງານບໍລິການເຮືອນພັກມີ 8 ທ່ານ ແລະ ບັນດາບໍລິສັດ ທີ່ ເຂົ້າມາດຳເນີນທຸລະກິດ ໃນສະຖານນີຂົນສົ່ງໂດຍສານປະກອບມີ: ບໍລິສັດ ຂົນສົ່ງໂດຍສານຈິດປະສົງ ຍອດນິຍົມ, ບໍລິສັດ ແສງສົມບູນ ຂົນສົ່ງໂດຍສານ, ບໍລິສັດ ແສນສະບາຍ ຂົນສົ່ງໂດຍສານ, ບໍລິສັດ ຈັນທະຈອນ ຂົນສົ່ງໂດຍສານ, ບໍລິສັດ ຈຳປາສັກ ຂົນສົ່ງໂດຍສານ, ບໍລິສັດ ແສງຈະເລີນ ລົດຕຽງ ນອນ, ບໍລິສັດ ກຽງໄກ VIP, ບໍລິສັດ ສີທອນ ພວງປະເສີດ ລົດຕຽງນອນ. ນອກຈາກນີ້ສະຖານນີຍັງ ມີສະຖານທີ່ພັກ , ຮ້ານຄ້າ, ຮ້ານຂາຍຍ່ອຍ, ຮ້ານອາຫານ ແລະ ສິ່ງອຳນວຍຄວາມສະດວກຕ່າງໆ ໄວ້ ເພື່ອບໍລິການຜູ້ໂດຍສານທີ່ມາລໍຖ້າລົດໄປຈຸດໝາຍປາຍທາງ.

#### 1.2 ຈຸດປະສົງຂອງການສຶກສາ

ວັດຖຸປະສົງຫຼັກຂອງບົດຈົບຊັ້ນໃນຄັ້ງນີ້ແມ່ນອີງໃສ່ໃນການພັດທະນາ ແລະ ເພີ່ມປະສິດທິພາບ ໃນການບໍລິການຂອງສະຖານນີຂົນສົ່ງໂດຍສານສາຍໃຕ້, ພວກຂ້າພະເຈົ້າຈຶ່ງໄດ້ມີແນວຄິດທີ່ຈະ ສ້າງ ແລະ ພັດທະນາ ລະບົບຈອງປີ້ລົດເມສາຍໃຕ້ອອນລາຍຂຶ້ນມາເພື່ອເກັບໄຂບັນຫາໃຫ້ກັບທາງ

ສະຖານນີຕາມວຽກຂອງແຕ່ລະພະແນກເຖິງບັນຫາໃນການເຮັດວຽກ ດັ່ງນັ້ນ, ພວກຂ້າພະເຈົ້າຈຶ່ງໄດ້ກຳນົດເຫດຜົນຂອງການຄົ້ນຄວ້າຕາມຈຸດປະສົງດັ່ງນີ້:

- ເພື່ອສຶກສາບັນຫາທີ່ເກີດຂຶ້ນໃນປະຈຸບັນ ແລະ ຄວາມຕ້ອງການຂອງລະບົບ.
- ເພື່ອສ້າງລະບົບຂາຍປີ້ລົດອອນໄລຂອງສະຖານນີຂົນສົ່ງໂດຍສານສາຍໃຕ້.
- ເພື່ອສ້າງຮູບແບບການຈັດການຂໍ້ມູນການໃຫ້ບໍລິການ.
- ເພື່ອເຜີຍແຜ່ຂໍ້ມູນການຂາຍປີ້ລົດເມຂອງສະຖານນີຂົນສົ່ງໂດຍສານສາຍໃຕ້.
- ເພື່ອການລາຍງານໃຫ້ສະດວກ ແລະ ຖືກຕ້ອງ.

### 1.3 ຂອບເຂດການຄົ້ນຄວ້າ

ລະບົບຈອງປີ້ລົດເມສາຍໃຕ້ແບບອອນລາຍ ເປັນລະບົບແບບ Client-Server Web Application ເຊິ່ງປະກອບດ້ວຍໜ້າວຽກຫຼັກດັ່ງນີ້:

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

### 1.4 ປະໂຫຍດທີ່ຄາດວ່າຈະໄດ້ຮັບ

ໃນການຂຽນບົດຈົບຊັ້ນໃນຄັ້ງນີ້ ຫຼັງຈາກສຳເລັດໂຄງການນີ້ແລ້ວ Web Application ແມ່ນຕ້ອງໃຫ້ໄດ້ຮັບຜົນຕົວຈິງ ແລະ ສາມາດນຳເຂົ້າມາໃຊ້ໃນວຽກງານຕົວຈິງໄດ້ຢ່າງແນ່ນອນ.

- ໄດ້ລະບົບຈອງປີ້ລົດເມແບບອອນລາຍຂອງສະຖານນີຂົນສົ່ງໂດຍສານສາຍໃຕ້.
- ໄດ້ລະບົບທີ່ຈະຊ່ວຍແກ້ໄຂບັນຫາການຈອງໄດ້ສະດວກ ແລະ ວ່ອງໄວຂຶ້ນ.
- ໄດ້ລະບົບຊ່ວຍເພີ່ມຊ່ອງທາງໃນການຂາຍປີ້ໃຫ້ກັບຜູ້ປະກອບການ.
- ມີລະບົບເຜີຍແຜ່.
- ໄດ້ລະບົບທີ່ສາມາດສ້າງລາຍງານໄດ້ຢ່າງສະດວກ ແລະ ຖືກຕ້ອງ.

## ບົດທີ 2

### ທົບທວນເອກະສານ ແລະ ບົດຄົ້ນຄວ້າທີ່ກ່ຽວຂ້ອງ

#### 2.1 ທົບທວນທິດສະດີທີ່ກ່ຽວຂ້ອງ

##### 2.1.1 ຄວາມຮູ້ພື້ນຖານກ່ຽວກັບຖານຂໍ້ມູນ

##### 2.1.1.1 ຄວາມໝາຍຂອງຖານຂໍ້ມູນ

Databases ຫຼື ຖານຂໍ້ມູນຄືກຸ່ມຂອງຂໍ້ມູນທີ່ຖືກເກັບລວບລວມໄວ້ໂດຍມີຄວາມສໍາພັນເຊິ່ງກັນ ແລະ ກັນໂດຍບໍ່ໄດ້ບັງຄັບວ່າຂໍ້ມູນທັງຫມົດນີ້ຈະຕ້ອງເກັບໄວ້ໃນແຟມຂໍ້ມູນດຽວກັນ ຫຼື ແຍກເກັບຫຼາຍໆ ແຟມຂໍ້ມູນ.

ລະບົບຖານຂໍ້ມູນຄືລະບົບທີ່ລວບລວມຂໍ້ມູນຕ່າງໆທີ່ກ່ຽວຂ້ອງກັນເຂົ້າໄວ້ດ້ວຍກັນຢ່າງມີລະບົບ, ມີຄວາມສໍາພັນລະຫວ່າງຂໍ້ມູນຕ່າງໆທີ່ຊັດເຈນ ໃນລະບົບຖານຂໍ້ມູນຈະປະກອບດ້ວຍແຟມຂໍ້ມູນຫຼາຍ ແຟມທີ່ມີຂໍ້ມູນກ່ຽວຂ້ອງກັນ, ສໍາພັນກັນເຂົ້າໄວ້ດ້ວຍກັນຢ່າງເປັນລະບົບ ແລະ ເປີດໂອກາດໃຫ້ຜູ້ໃຊ້ ສາມາດໃຊ້ງານ ແລະ ຮັກສາປ້ອງກັນຂໍ້ມູນເຫຼົ່ານີ້ໄດ້ຢ່າງມີປະສິດທິພາບ ໂດຍມີຊອບແວທີ່ປຸງປະ ເຫມືອນຊື່ກາງລະຫວ່າງຜູ້ໃຊ້ ແລະ ໂປຣແກຣມຕ່າງໆທີ່ກ່ຽວຂ້ອງກັບການໃຊ້ຖານຂໍ້ມູນເອີ້ນວ່າລະບົບ ຈັດການຖານຂໍ້ມູນ ຫຼື DBMS (Databases Management System) ມີຫນ້າທີ່ຊ່ວຍໃຫ້ຜູ້ໃຊ້ເຂົ້າເຖິງຂໍ້ ມູນໄດ້ງ່າຍສະດວກ ແລະ ມີປະສິດທິພາບການເຂົ້າເຖິງຂໍ້ມູນຂອງຜູ້ໃຊ້ອາດເປັນການສ້າງຖານຂໍ້ມູນ, ການແກ້ໄຂຖານຂໍ້ມູນ ຫຼື ການຕັ້ງຄໍາຖາມເພື່ອໃຫ້ໄດ້ຂໍ້ມູນມາໂດຍຜູ້ໃຊ້ບໍ່ຈໍາເປັນຕ້ອງຮັບຮູ້ກ່ຽວກັບ ລາຍລະອາດພາຍໃນໂຄງສ້າງຂອງຖານຂໍ້ມູນ.

##### 2.1.1.2 ຄຸນລັກສະນະຂອງຖານຂໍ້ມູນ

- ລົດຄວາມຊໍ້າຊ້ອນຂອງຖານຂໍ້ມູນໃຫ້ເຫຼືອນ້ອຍທີ່ສຸດ (Minimum Redundancy)
- ມີຄວາມຖືກຕ້ອງສູງສຸດ (Maximum Integrity)
- ມີຄວາມເປັນອິດສະຫຼະຂອງຂໍ້ມູນ (Data Independence)
- ມີລະບົບຄວາມປອດໄພຂອງຂໍ້ມູນສູງ (High Degree of Data Security)
- ການຄວບຄຸມຖານຂໍ້ມູນຈະຢູ່ສ່ວນກາງ (Logically Centralized Control)

##### 2.1.1.3 ຄວາມສໍາຄັນຂອງລະບົບຖານຂໍ້ມູນ

- ສາມາດລົດຄວາມຊໍ້າຊ້ອນຂອງຂໍ້ມູນໄດ້.
- ຫຼີກລ້ຽງຄວາມຂັດແຍ້ງຂອງຂໍ້ມູນໄດ້.
- ສາມາດໃຊ້ຂໍ້ມູນຮ່ວມກັນໄດ້.
- ສາມາດກໍານົດຄວາມເປັນມາດຕະຖານດຽວກັນຂອງຂໍ້ມູນໄດ້.
- ສາມາດກໍານົດລະບົບຄວາມປອດໄພຂອງຂໍ້ມູນໄດ້.

#### 2.1.1.4 ສະຖາປັດຕະຍະກຳລະບົບຖານຂໍ້ມູນ

ສະຖາປັດຕະຍະກຳລະບົບຖານຂໍ້ມູນ ມີ 3 ລະດັບຄື:

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

##### 1) ລະດັບພາຍໃນ (Internal Level)

ເປັນການນຳເອົາຂໍ້ມູນທີ່ໄດ້ຈາກລະດັບແນວຄິດມາວິເຄາະ ແລະ ອອກແບບໂດຍແປງໃຫ້ຢູ່ໃນຮູບແບບຂອງການຈັດການຖານຂໍ້ມູນ (DBMS) ທີ່ເລືອກໃຊ້ໂດຍເສີມແນວຄິດການເຮັດ Normalization ແລະ Demoralization.

##### 2) ລະດັບລັກສະນະແນວຄິດ (Conceptual Level)

ເປັນການນຳເອົາຂໍ້ມູນທີ່ໄດ້ຈາກການວິເຄາະຄວາມຕ້ອງການຂອງຜູ້ໃຊ້ຂໍ້ມູນໃນລະດັບພາຍນອກມາອອກແບບຖານຂໍ້ມູນ ເພື່ອໄດ້ໂຄງຮ່າງຂອງຖານຂໍ້ມູນໃນລະດັບແນວຄິດທີ່ປະກອບດ້ວຍໂຄງສ້າງຂອງຖານຂໍ້ມູນສ່ວນໜຶ່ງເປັນຄວາມສຳພັນກັນ.

##### 3) ລະດັບພາຍນອກ (External Level)


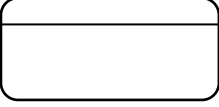

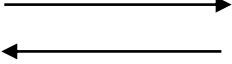

ເປັນການນຳເອົາຂໍ້ມູນທີ່ໄດ້ຈາກລະດັບພາຍໃນມາກຳນົດໂຄງສ້າງຂໍ້ມູນ ແລະ ການຈັດເກັບວິທີການເຂົ້າເຖິງການຈັດການດ້ານລະບົບຄວາມປອດໄພຂອງຂໍ້ມູນເພື່ອຖານຂໍ້ມູນເຮັດວຽກໄດ້ຢ່າງມີປະສິດທິພາບ.

#### 2.1.2 ທິດສະດີທີ່ກ່ຽວຂ້ອງກັບ DFD (Data Flow Diagram)

##### 2.1.2.1 ຈຸດປະສົງຂອງ DFD

- ເປັນແຜນພາບທີ່ສະຫຼຸບລວມຂໍ້ມູນທັງໝົດໄດ້ຈາກການວິເຄາະໃນລັກສະນະຂອງຮູບແບບທີ່ເປັນໂຄງສ້າງ.
- ເປັນຂໍ້ຕົກລົງຮ່ວມກັນລະຫວ່າງນັກວິເຄາະລະບົບ ແລະ ຜູ້ຊົມໃຊ້.
- ເປັນແຜນພາບທີ່ໃຊ້ໃນການພັດທະນາຕໍ່ໃນຂັ້ນຕອນຂອງການອອກແບບ.
- ຮູ້ທີ່ໄປທີ່ມາຂອງຂໍ້ມູນທີ່ໄຫຼໃນຂະບວນການຕ່າງໆ.

##### 2.1.2.2 ສັນຍະລັກທີ່ໃຊ້ໃນແຜນວາດການໄຫຼຂໍ້ມູນ

ຊື່	ສັນຍາລັກ	ຄວາມໝາຍ
Boundary Or External Entity		ຂອບເຂດໝາຍເຖິງພາກສ່ວນທີ່ກ່ຽວຂ້ອງກັບລະບົບ ເຊິ່ງລະບົບບໍ່ສາມາດຄວບຄຸມໄດ້
Process		ປະມວນຜົນ ຫຼື ຫນ້າວຽກທີ່ເຮັດໃນໂຄງການນັ້ນໆ
Data Store		ບ່ອນຈັດເກັບຂໍ້ມູນ
Data Flow		ການໄຫຼຂອງຂໍ້ມູນ
Real-Time Link		ການເຊື່ອມໂຍງແບບໄກທີ່ມີການຕອບກັບແບບທັນທີທັນໃດ

ຕາຕະລາງທີ 2 ສະແດງສັນຍາລັກ **Data Flow Diagram**

### 2.1.2.3 ກົດຂອງ Process

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

### 3.1.3.4 ຫຼັກການຂຽນແຜນວາດການໄຫຼຂໍ້ມູນ

#### 1) Process:

- ເມື່ອມີຂໍ້ມູນເຂົ້າໄປທີ່ Process ກໍ່ຕ້ອງມີຂໍ້ມູນ ຫຼື ຜົນຮັບອອກມາຈາກ Process ເຊັ່ນກັນຈະເປັນໄປບໍ່ໄດ້ທີ່ມີສະເພາະຂໍ້ມູນເຂົ້າຢ່າງດຽວ.

2) Data store:

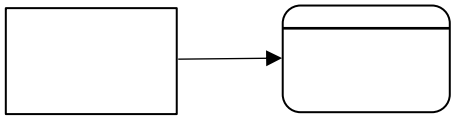
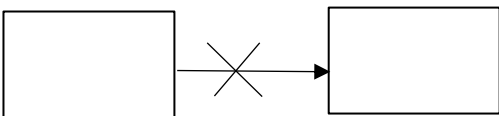
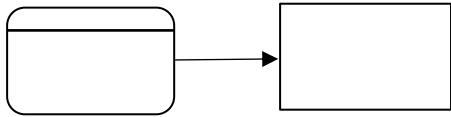
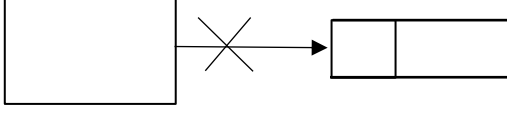
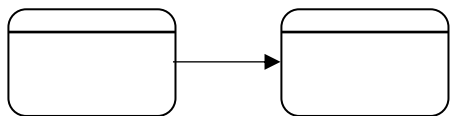
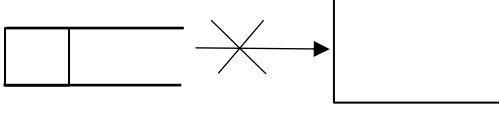
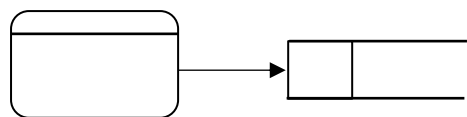
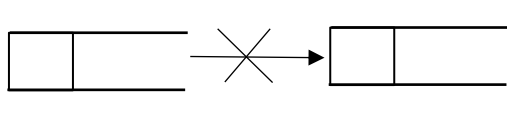
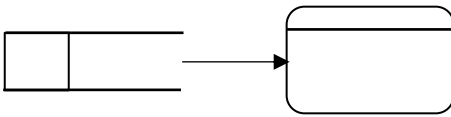
- ຂໍ້ມູນຈະໄຫຼຈາກ Data store ໜຶ່ງໄປຫາ Data store ໜຶ່ງໂດຍກົງບໍ່ໄດ້ຈະຕ້ອງຜ່ານ Process ເທົ່ານັ້ນ.
- ຂໍ້ມູນທີ່ສົ່ງຜ່ານ External entity ບໍ່ສາມາດໄຫຼເຂົ້າໄປ Data store ໂດຍກົງໄດ້ ຈະຕ້ອງໃຊ້ Process ເປັນຕົວກາງໃນການເຊື່ອມໂຍງເພື່ອຈັດເກັບຂໍ້ມູນໃນ Data store.
- ຂໍ້ມູນທີ່ໄຫຼຜ່ານຈາກ Data store ບໍ່ສາມາດເຊື່ອມໂຍງເຂົ້າກັບ External entity ໄດ້ໂດຍກົງຈະຕ້ອງຜ່ານ Process ເທົ່ານັ້ນ.

3) External entity:

- External entity ບໍ່ສາມາດເຊື່ອມໂຍງເຂົ້າຫາກັນໄດ້ຈະຕ້ອງໃຊ້ Process ເປັນຕົວກາງເພື່ອສົ່ງຜ່ານ ແລະ ຊື່ຂອງ External entity ຈະໃຊ້ຄຳນາມເທົ່ານັ້ນ.

4) Data flow:

- ການໄຫຼຂໍ້ມູນທີ່ມີຫົວຊີ້ໄປທີ່ Process ໝາຍເຖິງ Process ມີການອ່ານ ຫຼື ການດຶງຂໍ້ມູນຈາກ Data store ມາໃຊ້ວຽກ.
- ການໄຫຼຂໍ້ມູນຈາກ Process ທີ່ມີຫົວລູກສອນຊີ້ໄປຍັງ Data store ໝາຍເຖິງການ Update ຫຼື ການເພີ່ມຂໍ້ມູນລົງໄປທີ່ Data store.
- ການໄຫຼຂໍ້ມູນທີ່ມີຫົວລູກສອນທັງສອງດ້ານທີ່ເຊື່ອມໂຍງລະຫວ່າງ Process ກັບ Data store ໝາຍເຖິງມີການດຶງຂໍ້ມູນຈາກ Data store ມາປັບປຸງ ແລະ ມີການ Update ຂໍ້ມູນລົງໄປໃນ Data store.
- ການໄຫຼຂໍ້ມູນບໍ່ສາມາດຍ້ອນກັບໄປຍັງ Process ເດີມໄດ້ ຢ່າງໜ່ວຍຕ້ອງເຊື່ອມໂຍງຜ່ານ Process ໜຶ່ງເພື່ອສົ່ງຜ່ານຍ້ອນກັບມາຍັງ Process ເດີມ ແລະ ຊື່ທີ່ລະບຸໃນການໄຫຼຂໍ້ມູນຈະໃຊ້ຄຳນາມ.

ອະນຸຍາດ	ບໍ່ອະນຸຍາດ
	
	
	
	
	

ຕາຕະລາງທີ 10 ຮູບການປຸງປະໂຫຍດແຜນວາດການໄຫຼຂໍ້ມູນທີ່ບໍ່ຖືກຕ້ອງ ແລະ ຖືກຕ້ອງ




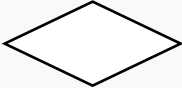



### 2.1.3 ຄວາມໝາຍ ແລະ ສັນຍະລັກຂອງ Flowchart

ສັນຍະລັກ Flowchart ຄືຮູບພາບທີ່ໃຊ້ແທນຄວາມໝາຍການເຮັດວຽກງານໃນລັກສະນະຕ່າງໆ ພາຍໃນແຜນຜັງ (Flowchart) ປະກອບໄປດ້ວຍ ການເລີ່ມຕົ້ນ (Start), ການຈົບ (End), ການກະທຳ (Process), ການນຳເຂົ້າຂໍ້ມູນ (Input), ການສະແດງຜົນຂໍ້ມູນ (Output), ການຕັດສິນໃຈ (Decision), ຄຳອະທິບາຍ (Annotation), ຈຸດເຊື່ອມຕໍ່ (Connector), ທິດທາງການເຮັດວຽກງານ (Direction Flow).

ສັນຍະລັກເຫຼົ່ານີ້ເມື່ອຖືກນຳມາເຊື່ອມຕໍ່ກັນ ຈະກາຍເປັນ "ແຜນຜັງ (Flowchart)" ທີ່ສະແດງລຳດັບຂັ້ນຕອນການເຮັດວຽກງານເພື່ອ

- ເປັນເຄື່ອງມືໃນການຈັດລຳດັບຄວາມຄິດ.
- ເຫັນລຳດັບຂັ້ນຕອນການເຮັດວຽກງານທີ່ຊັດເຈນ.

- ສັນຍະລັກ Flowchart

ຮູບພາບສັນຍະລັກ	ຄວາມໝາຍຂອງສັນຍະລັກ
	ການເລີ່ມຕົ້ນ ຫຼື ຈົບ Flowchart (Start ຫຼື End)
	ການກະທຳ (Process) ຖືກໃຊ້ເພື່ອສະແດງທີ່ການກະທຳໃນ Flowchart
	ຮັບຂໍ້ມູນ
	ການຕັດສິນໃຈ (Decision)
	ສະແດງຜົນທາງຈຳພາບ
	ຈຸດເຊື່ອມຕໍ່ (Connector)
	ທິດທາງການເຮັດວຽກງານ (Direction Flow)

#### 2.1.4 ແຜນວາດຄວາມສຳພັນຂອງ Entity (ER Diagram)

ການອອກແບບການສ້າງຕາຕະລາງຖານຂໍ້ມູນແບບຈຳລອງ ER ແບ່ງອອກເປັນ 2 ຂັ້ນຕອນຫຼັກຄື: ຂັ້ນຕອນທຳອິດເປັນການສ້າງແບບຈຳລອງ ER ຂັ້ນຕອນນີ້ຈະກ່າວເຖິງຄວາມໝາຍຂອງສັນຍາລັກຕ່າງໆທີ່ໃຊ້ໃນແບບຈຳລອງ ER, ຂັ້ນຕອນທີ 2 ແມ່ນການແປງແບບຈຳລອງ ER ໃຫ້ເປັນຕາຕະລາງຂໍ້ມູນເພື່ອໃຊ້ເປັນຕົວຈັດການຂໍ້ມູນ. ເມື່ອເຂົ້າໃຈໃນທັງສອງແບບນີ້ແລ້ວສາມາດອອກແບບຖານຂໍ້ມູນ ແລະ ສ້າງຖານຂໍ້ມູນໃນແບບຈຳລອງ ER ຂຶ້ນມາໃຊ້ງານດ້ວຍຕົວເອງ.

##### 2.1.4.1 ສັນຍາລັກທີ່ໃຊ້ໃນແບບຈຳລອງ ER

ການອອກແບບຈຳລອງຖານຂໍ້ມູນ ER ນັ້ນກ່ອນອື່ນຕ້ອງຮູ້ຈັກຄວາມໝາຍສັນຍາລັກຕ່າງໆທີ່ໃຊ້ໃນການອອກແບບຈຳລອງ ER ໄດ້ຢ່າງຖືກຕ້ອງ ດັ່ງນັ້ນ ໃນຫົວຂໍ້ນີ້ຈະເວົ້າເຖິງຄວາມໝາຍ ແລະ ການໃຊ້ງານສັນຍາລັກຕ່າງໆຂອງ ER.

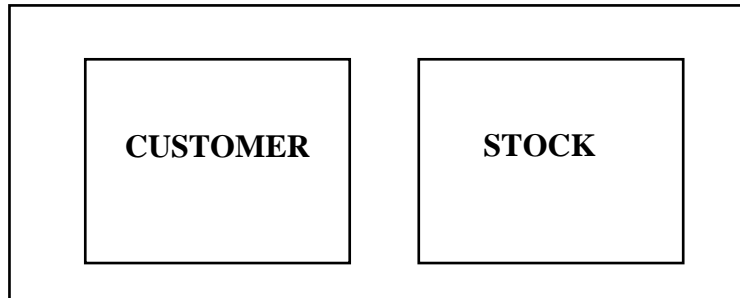


### 1) ເອັນຕິຕີ (Entity)

ເອັນຕິຕີ ຄືວັດຖຸທີ່ເຮົາສົນໃຈເຊິ່ງອາດເປັນໄດ້ທັງບຸກຄົນ, ສະຖານທີ່, ວັດຖຸ, ເຫດການ ຫຼື ແນວຄິດທີ່ກໍ່ໃຫ້ເກີດກຸ່ມຂອງຂໍ້ມູນທີ່ຕ້ອງການເອັນຕິຕີແບ່ງອອກເປັນ 2 ປະເພດຄື:

### 2) Strong Entity:

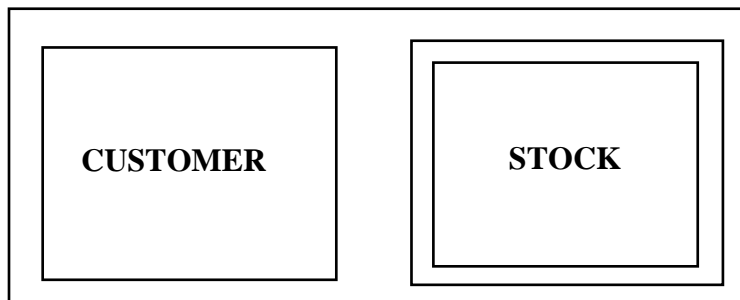
ເປັນເອັນຕິຕີທີ່ເກີດຂຶ້ນດ້ວຍຕົນເອງເປັນອິດສະຫຼະບໍ່ຂຶ້ນກັບເອັນຕິຕີໃດສັນຍາລັກທີ່ໃຊ້ຄືຮູບສີ່ຫຼ່ຽມ ແລະ ສາມາດເອີ້ນ Strong Entity ໄດ້ອີກຊື່ໜຶ່ງວ່າ Regular Entity.



ຮູບທີ 3 ຮູບ Strong Entity

### 3) Weak Entity:

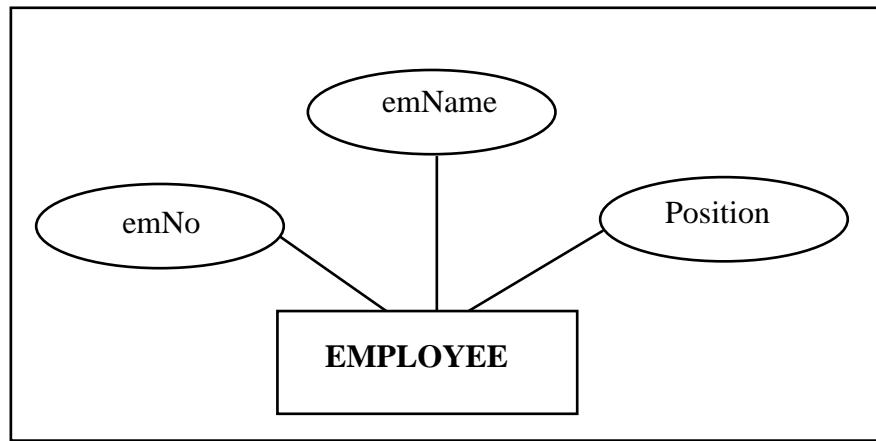
ເອັນຕິຕີຊະນິດນີ້ຈະຂຶ້ນກັບເອັນຕິຕີຊະນິດອື່ນໆບໍ່ສາມາດເກີດຂຶ້ນໄດ້ຕາມລຳພັງ ແລະ ຈະຖືກລົບເມື່ອເອັນຕິຕີຫຼັກຖືກລົບອອກ ສັນຍາລັກທີ່ໃຊ້ຄືຮູບສີ່ຫຼ່ຽມຊ້ອນກັນ.



ຮູບທີ 4 ຮູບ Weak Entity

### 4) ແອັດທິບິວ (Attribute)

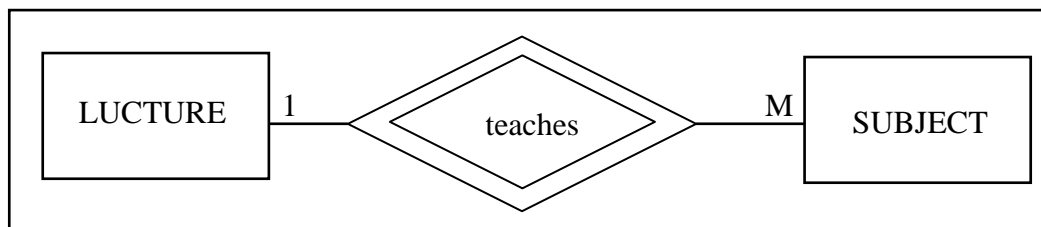
ແອັດທິບິວຄືຄຸນສົມບັດຂອງສົມບັດຂອງເອັນຕິຕີສັນຍາລັກຂອງແອັດທິບິວຈະເປັນຮູບວົງນິ້ແອັດທິບິວໃດທີ່ຖືກໃຊ້ເປັນຄືຫຼັກຈະຖືກຂີດເສັ້ນກ້ອງກຳກັບໄວ້.



ຮູບທີ 5 ຮູບAttribute

#### 5) ຄວາມສຳພັນ (Relation)

ຄວາມສຳພັນໃນທີ່ນີ້ ໝາຍເຖິງຄວາມສຳພັນລະຫວ່າງເອ້ນຕີຕີ ໂດຍແຕ່ລະຄວາມສຳພັນຄວນມີ ຊື່ລະບຸໄວ້ເພື່ອໃຊ້ອະທິບາຍເຊິ່ງປົກກະຕິຈະໃຊ້ສັນຍາລັກຮູບດອກຈັນທີ່ພາຍໃນລະບຸຄຳກິລິຍາໄວ້ເພື່ອ ອະທິບາຍຄວາມສຳພັນ.



ຮູບທີ 6 ຮູບ Relation

#### 2.1.5 ຄວາມຮູ້ພື້ນຖານກ່ຽວກັບພາສາ NOSQL

ຫຼາຍຄົນຄົງພໍຈະເຄີຍໄດ້ຍິນກັນມາແຕ່ກ່ຽວກັບເທັກໂນໂລຢີ ການຈັດການຂໍ້ມູນແບບໃໝ່ນີ້ ຊຶ່ງ ກໍຄື NoSQL ເມື່ອເວົ້າເຖິງ NoSQL ຈະໄດ້ຍິນຊື່ເວັບໄຊທີ່ໃຫຍ່ໆ ເຊັ່ນ Facebook, Twitter, FourSquare, Digg ແລະ ອື່ນໆ. ເຮັດໃຫ້ເຮົາຮັບຮູ້ວ່າ NoSQL ເປັນລະບົບຖານຂໍ້ມູນສຳລັບງານທີ່ ຕ້ອງຮອງຮັບຂໍ້ມູນຂະໜາດໃຫຍ່ໆ ຮອງຮັບການຂະຫຍາຍລະບົບໄດ້ງ່າຍເປັນຕົ້ນ.

ຊຶ່ງກໍເປັນເຊັ່ນນັ້ນແທ້ ແຕ່ວຽກທີ່ນ້ອຍໆຈະເຮັດຢ່າງໃດໃຊ້ງານໄດ້ບໍ່ ຄຸ້ມຄ່າທີ່ຈະນຳ NoSQL ມາໃຊ້ງານຂະໜາດນ້ອຍ ຫຼື ບໍ່ ຫຼືໃຊ້ Relational Database ກໍພຽງພໍແລ້ວ ຄຳຕອບຄືຂຶ້ນຢູ່ກັບລັກສະນະ ໃນການນຳມາໃຊ້ງານ ກ່ອນທີ່ຈະຕອບຄຳຖາມວ່າ NoSQL ເປັນຄຳຕອບຂອງລະບົບຈັດເກັບຂໍ້ມູນຂອງ ຫຼື ບໍ່ ລອງພິຈາລະນາຫົວຂໍ້ຕ່າງໆດັ່ງຕໍ່ໄປນີ້:

### 1) ຜູ້ໃຊ້ນັບມື້ນັບຫຼາຍ (BigUsers)

ຈະເຫັນໄດ້ວ່າໃນຊ່ວງເວລາທີ່ຜ່ານມານີ້ ແລະ ໃນປັດຈຸບັນຜູ້ທີ່ໃຊ້ງານ Internet ມີແນວໂນ້ມ ຫຼາຍຂຶ້ນເລື້ອຍໆ ບໍ່ວ່າຈະໃຊ້ງານຜ່ານ Desktop PC ຫຼື Smartphone ຊຶ່ງເທັກໂນໂລຊີຂອງອຸປະກອນ (Devices) ມີຫຼາຍຂຶ້ນ ແລະ ໃຊ້ງານໄດ້ງ່າຍຂຶ້ນ.

ການພັດທະນາລະບົບສາມາດຮອງຮັບປະລິມານການເຂົ້າໃຊ້ງານແຕ່ລະອຸປະກອນ (Devices) ເປັນສິ່ງໜຶ່ງທີ່ຕ້ອງນຳມາພິຈາລະນາ ແລະ ບໍ່ພຽງແຕ່ຕ້ອງຮອງຮັບການເຂົ້າໃຊ້ງານຂອງຜູ້ໃຊ້ໄດ້ເທົ່ານັ້ນ ເຮົາຕ້ອງຮອງຮັບວິທີການປ້ອນຂໍ້ມູນແບບໃໝ່ຄືແຕ່ກ່ອນຜູ້ຈັດການເນື້ອຫາຕ່າງໆ ຄື Web Master, Web Editor, ຜູ້ເບິ່ງແຍງລະບົບເປັນຕົ້ນ. ແຕ່ປະຈຸບັນຜູ້ທີ່ປ້ອນຂໍ້ມູນຄືຜູ້ໃຊ້ບໍລິການ (users) ໂດຍກົງ ຜ່ານອຸປະກອນ (Devices) ຕ່າງໆທີ່ມີຫຼາກຫຼາຍ ແລະ ການປ້ອນຂໍ້ມູນກໍ່ງ່າຍກວ່າແຕ່ກ່ອນອີກດ້ວຍ.

ຍັງມີປັດໄຈອື່ນໆເຊັ່ນ ເທດສະການສຳຄັນໆທີ່ຄົນຈະເຂົ້າມາໃຊ້ງານຫຼາຍເປັນພິເສດ ຫຼື ຜູ້ໃຊ້ ງານທີ່ບໍ່ແມ່ນແຕ່ປະເທດເຮົາເທົ່ານັ້ນ ເພາະໂລກອິນເຕີເນັດເຖິງກັນ ອາດຈະຕ້ອງເບິ່ງວ່າລະບົບເຮົາມີຜູ້ ເຂົ້າໃຊ້ງານຈາກຕ່າງປະເທດ ຫຼື ທົ່ວໂລກ ຫຼື ບໍ່ອີກດ້ວຍ.

ດັ່ງນັ້ນ ເຮົາຕ້ອງກັບມາທົບທວນວິທີການຈັດການຖານຂໍ້ມູນແລ້ວວ່າ ບໍ່ແມ່ນແຕ່ເຮັດໃຫ້ຮອງ ຮັບກັບການເຂົ້າມາໃຊ້ງານຂອງຜູ້ໃຊ້ບໍລິການເທົ່ານັ້ນ ແຕ່ຕ້ອງຮອງຮັບການຈັດເກັບຂໍ້ມູນທີ່ຫລາຍຂຶ້ນ ເລື້ອຍໆໄດ້ອີກ.

### 2) ປະເພດຂໍ້ມູນຕ່າງໆ ແລະ ຂໍ້ມູນທີ່ຕ້ອງການຈັດເກັບຫຼາຍຂຶ້ນເລື້ອຍໆ (BigData)

ຈາກຕົວແປຂອງຜູ້ໃຊ້ງານມີຫຼາຍຂຶ້ນເລື້ອຍໆ ອຸປະກອນໃນການເຂົ້າໃຊ້ງານກໍ່ຫຼາກຫຼາຍປະເພດ ຂອງຂໍ້ມູນທີ່ໄດ້ຈາກແຕ່ລະອຸປະກອນກໍ່ຫຼາກຫຼາຍປະເພດ ເຊັ່ນ ຂໍ້ຄວາມ, ຮູບພາບ, ສຽງ, ວິດີໂອ, ຕຳ ແໜ່ງສະຖານທີ່ (GeoLocation) ແລະ ອື່ນໆ ແລະ ການປ້ອນຂໍ້ມູນເຫຼົ່ານີ້ກໍ່ງ່າຍແສນໆ ເພາະເທັກ ໂນໂລຢີຂອງຮາດແວຣ໌ ແລະ ຊອບແວຣ໌ມີການພັດທະນາຫຼາຍຂຶ້ນເລື້ອຍໆ ໃຊ້ງານງ່າຍຂຶ້ນສະດວກຂຶ້ນ ວ່ອງໄວຂຶ້ນເລື້ອຍໆ.

ດັ່ງນັ້ນ ການຈັດເກັບຂໍ້ມູນທີ່ລ້ຽງໄຫຼເຂົ້າມາຈາກອຸປະກອນຕ່າງໆເຫຼົ່ານີ້ ເຮົາອາດຈະຕ້ອງນຳມາວິ ຄາະພຶດຕິກຳຂອງຜູ້ໃຊ້ບໍລິການ ການສົ່ງເສີມການຕະຫຼາດ ເຮັດຂໍ້ມູນການຕັດສິນໃຈຂອງຜູ້ບໍລິຫານ ຂໍ ມູນລູກຄ້າສຳພັນ ແລະ ອື່ນໆອີກຫຼວງຫລາຍການນຳລະບົບຖານຂໍ້ມູນແບບເດີມ(Relational Database) ອາດຈະບໍ່ເໝາະກັບລັກສະນະງານບາງຢ່າງອີກຕໍ່ໄປ.

### 3) ເຕັກໂນໂລຢີຮາດແວໄດ້ປ່ຽນແປງລາຄາຖືກຫຼຸດລົງແຕ່ວ່າປະສິດທິພາບດີຂຶ້ນ (Cloud Computing).

ເຮົາອາດຈະເລີ່ມໄດ້ຍິນຄຳວ່າ Cloud Technology ຊຶ່ງກໍມີຫຼາຍປະເພດ ແຕ່ໃນນີ້ເຮົາຈະເວົ້າເຖິງໃນແງ່ຂອງການນຳມາໃຊ້ງານໂດຍປະຈຸບັນຖ້າໃຜເຄີຍໄດ້ລອງໃຊ້ງານ EC2 ຂອງ Amazon ມາແດ່ແລ້ວຈະຮູ້ຈັກເປັນຢ່າງດີວ່າການຈະມີເຄື່ອງ Server ແຮງໆຈັກເຄື່ອງ ເປັນເລື່ອງທີ່ງ່າຍຫຼາຍ ຫຼື ການຈະມີ Server 10 ເຄື່ອງ ຫຼື 20 ເຄື່ອງ ນຳມາຕໍ່ເປັນ Database Cluster ນັ້ນງ່າຍຫຼາຍພຽງແຕ່ຄລິກສ້າງ Instance ບໍ່ຈັກເທື່ອກໍໄດ້ Server ມາໃຊ້ງານແລ້ວ ແລະ ລາຄາກໍຖືກຫຼາຍ ຖ້າເຮົາບໍ່ໃຊ້ງານແລ້ວກໍຍົກເລີກການໃຊ້ງານ ແລະ ຄືນກັບໄປໄດ້ທັນທີ ເມື່ອທຽບກັບສະໄໝກ່ອນທີ່ເຮົາຕ້ອງການມີ Server ຈັກ 10 ເຄື່ອງ ຈະຕ້ອງລົງທຶນຊື້ເຄື່ອງມາຫຼາຍ ຖ້າເຮົາໃຊ້ງານແລ້ວຕ້ອງແບກຮັບພາລະເຄື່ອງ Server ເຫຼົ່ານີ້ໄວ້ຊຶ່ງເປັນຕົ້ນທຶນທີ່ແພງຫຼາຍ.

ຈາກທີ່ກ່າວມາຂ້າງເທິງ ເຮົາບໍ່ໄດ້ເນັ້ນຂໍ້ດີຂອງ EC2 ແຕ່ຢ່າງໃດແຕ່ກຳລັງຈະເນັ້ນວ່າພາບລວມຂອງການໃຊ້ງານ Server ເລີ່ມປ່ຽນໄປຄືໃຊ້ງານໄດ້ງ່າຍຂຶ້ນ ລາຄາຖືກລົງ ແຕ່ປະສິດທິພາບດີຂຶ້ນ ຊຶ່ງເປັນສິ່ງທີ່ສຳຄັນໃນການນຳມາພິຈາລະນາຂອງເທັກໂນໂລຢີດ້ານຖານຂໍ້ມູນຄື ຖ້າຕ້ອງການຈັດເກັບຖານຂໍ້ມູນໃຫຍ່ໆ ຫຼື ຮອງຮັບຜູ້ໃຊ້ງານໄດ້ເປັນຈຳນວນຫຼາຍ ການຂະຫຍາຍລະບົບຖານຂໍ້ມູນເປັນເລື່ອງທີ່ງ່າຍຂຶ້ນ ຊຶ່ງເຮັດໄດ້ໂດຍການເອົາ Server ມາຕັ້ງກັນອອກໄປ ຫຼື ເອີ້ນວ່າການຂະຫຍາຍອອກແນວນອນ (Scale Out) ບໍ່ແມ່ນການຂະຫຍາຍລະບົບຄືແຕ່ກ່ອນ ຄືຂະຫຍາຍອອກແນວຕັ້ງ (Scale Up) ແລະ ຕ້ອງໃຊ້ເຄື່ອງ Server ທີ່ມີປະສິດທິພາບສູງ ຊຶ່ງຈະມີຕົ້ນທຶນທີ່ແພງກວ່າການຂະຫຍາຍແບບແນວນອນຫລາຍ.

ດັ່ງນັ້ນ ການຂະຫຍາຍລະບົບທີ່ຢູ່ເທິງພື້ນຖານຂອງ NoSQL ຄືຮອງຮັບການຂະຫຍາຍລະບົບແບບແນວນອນ (Scale Out) ຊຶ່ງຈະກະຈາຍຂໍ້ມູນໄປເກັບທີ່ເຄື່ອງ Server ຫຼາຍເຄື່ອງ ແລະ ໃຊ້ເຄື່ອງ Server ທົ່ວໄປທີ່ເອີ້ນວ່າ (Commodity Server) ບໍ່ຈຳເປັນຕ້ອງໃຊ້ Server ທີ່ເອີ້ນວ່າ Enterprise Server ທີ່ມີລາຄາແພງຕາມ Spec ທີ່ສູງຂຶ້ນເລື້ອຍໆ ແລະ ການບໍລິຫານຈັດການກໍຍາກຂຶ້ນອີກດ້ວຍ.

### 4) ບັນຫາຂອງ Relational Database

ຖ້າໃຜທີ່ໃຊ້ງານ Relation Database ທີ່ຕ້ອງການຮອງຮັບການຈັດເກັບຂໍ້ມູນຂະໜາດໃຫຍ່ ຕ້ອງຫຼີກລ້ຽງບໍ່ໄດ້ເລື່ອງການເຮັດ Sharding ແລະ Distributed Cache ເພາະເປັນຕົວຫຼັກທີ່ຕ້ອງເຮັດ ເພື່ອຂະຫຍາຍລະບົບຖານຂໍ້ມູນຂອງ Relational Database ໃຫ້ສາມາດຮອງຮັບຂໍ້ມູນທີ່ຫລາຍຂຶ້ນ ແລະ ຮອງຮັບຈຳນວນການເຂົ້າມາໃຊ້ງານລະບົບໄດ້ຫຼາຍຂຶ້ນ.

### 5) Manual Sharding

ການແບ່ງຕາຕະລາງຖານຂໍ້ມູນ (Table) ອອກເປັນສ່ວນ ແລ້ວກໍກະຈາຍໄປຈັດເກັບໃນຫຼາຍໆ Server ເພື່ອໃຫ້ແຕ່ລະຕາຕະລາງ (Table) ຂອງຖານຂໍ້ມູນບໍ່ຈັດເກັບຂໍ້ມູນທີ່ຫຼາຍເກີນໄປ ເພາະຖ້າຂໍ້

ມູນໃນແຕ່ລະຖານຂໍ້ມູນຫຼາຍເກີນໄປ ຈະເຮັດໃຫ້ລະບົບຖານຂໍ້ມູນຊ້າ ແຕ່ບັນຫາກໍຈະຕາມມາອີກຄື ເມື່ອຕ້ອງກະຈາຍຂໍ້ມູນອອກໄປໃນແຕ່ລະ Server ການຈະ ເກັບຂໍ້ມູນ ເຊັ່ນ: ເພີ່ມ, ແກ້ໄຂ, ລຶບ, ດຶງຂໍ້ມູນ ມາສະແດງຕ່າງໆ ຈະຕ້ອງເຮັດຜ່ານ Application ຫຼື ຕ້ອງມີ Server ບາງໂຕທີ່ຖືກດຶງຂໍ້ມູນແຕ່ລະ Server ມາທັງໝົດເປັນກ້ອນດຽວ ນັ້ນໝາຍຄວາມວ່າ ເຮົາຕ້ອງເຮັດດ້ວຍໂຕເຮົາເອງ ບໍ່ແມ່ນລະບົບຖານຂໍ້ມູນຈັດການໃຫ້ (Manual Sharding).

## 6) Distributed Cache

ເມື່ອເຮົາຕ້ອງການໃຫ້ລະບົບຮອງຮັບການເຂົ້າມາໃຊ້ງານຫຼາຍໄດ້ນັ້ນ ຖ້າຈະຕ້ອງເຂົ້າມາອ່ານຂໍ້ມູນຜ່ານ Database ໂດຍກົງມັນອາດຈະຮອງຮັບບໍ່ໄຫວ ຫຼື ເຮັດໄດ້ຊ້າ ດັ່ງນັ້ນ ຈະຕ້ອງມີການເຮັດ Cache Layer ຂຶ້ນມາ ຄືແທນທີ່ຈະເຂົ້າໄປອ່ານຈາກຖານຂໍ້ມູນໂດຍກົງ ກໍໃຫ້ອ່ານຜ່ານ Cache ກ່ອນ ດັ່ງນັ້ນການອ່ານຂໍ້ມູນຈາກ Cache ເປັນການອ່ານຈາກ Memory ໂດຍກົງ ເຮັດໃຫ້ຮອງຮັບປະລິມານການເຂົ້າມາໃຊ້ງານໄດ້ຫຼາຍຂຶ້ນ.

ແຕ່ບັນຫາຄືການເຮັດ Cache Layer ນີ້ຮອງຮັບສະເພາະການອ່ານຂໍ້ມູນເທົ່ານັ້ນ ບໍ່ຮອງຮັບການຂຽນຂໍ້ມູນໄດ້ ຖ້າຕ້ອງການຮອງຮັບການຂຽນຂໍ້ມູນປະລິມານຫຼາຍ ແລະ ອ່ານຂໍ້ມູນປະລິມານຫຼາຍ ຈຶ່ງເປັນສິ່ງທີ່ Relational Database ບໍ່ສາມາດຮອງຮັບງານໃນລັກສະນະ ອ່ານ, ຂຽນ ຂໍ້ມູນປະລິມານຫຼາຍໄດ້ຄື ແລະ ສິ່ງສໍາຄັນການເຮັດ Cache Layer ຈະຕ້ອງມີການດູແລຮັກສາ ແລະ ໃຊ້ Server ແຍກອອກໄປຕ່າງຫາກອີກ.

ຈາກຈຸດນີ້ເອງການເຮັດ Sharding ແລະ Caching ເປັນສິ່ງທີ່ຖືກພັດທະນາຂຶ້ນໃນ NoSQL ເທັກໂນໂລຢີ ໂດຍຮອງຮັບ Auto-Sharding ແລະ Integrated Caching ໃນຕົວເອງ ດັ່ງນັ້ນ ເຮົາຈຶ່ງໄດ້ເຫັນ NoSQL ຖືກນໍາໄປໃຊ້ງານກັບລະບົບໃຫຍ່ໆເຊັ່ນ: Facebook, Twitter, FourSquare, Digg ແລະ ອື່ນໆ ເພາະວ່າ NoSQL ອອກແບບມາເພື່ອຮອງຮັບຄວາມຕ້ອງການງານໃຫຍ່ໆໄດ້ດີໂດຍສະເພາະຢູ່ແລ້ວ ແຕ່ເຖິງຢ່າງໃດກໍຕາມຍັງມີຄຸນສົມບັດອື່ນໆ ທີ່ໜ້າສົນໃຈໃນ NoSQL ເທັກໂນໂລຢີ.

## ກ. ຄຸນສົມບັດຂອງ NoSQL Database

### - Dynamic Schemas

ການຈັດເກັບຂໍ້ມູນຕ່າງໆໃນຖານຂໍ້ມູນແບບ Relational Database ເຮົາຈະຕ້ອງມີການສ້າງ Schema ຫຼື ຮູບແບບຂອງໂຄງສ້າງຕາຕະລາງວ່າຈະຈັດເກັບຂໍ້ມູນຫຍັງ ເມື່ອຕ້ອງການຈັດເກັບຂໍ້ມູນເພີ່ມເຕີມຕ້ອງປ່ຽນ Schema ພາຍຫຼັງ (Alter-Table) ກ່ອນຈະຈັດເກັບຂໍ້ມູນຮູບແບບໃໝ່ໄດ້.

ແຕ່ໃນປະຈຸບັນການຈັດເກັບຂໍ້ມູນມີການປ່ຽນແປງຕະຫຼອດເວລາ ເພາະຄວາມຕ້ອງການຈັດເກັບຂໍ້ມູນຕ່າງໆມີຫຼາກຫຼາຍຂຶ້ນເລື້ອຍໆ ການກໍານົດໂຄງສ້າງຂອງຕາຕະລາງຖານຂໍ້ມູນ ຫຼື ການຕ້ອງປ່ຽນໂຄງສ້າງຖານຂໍ້ມູນເລື້ອຍໆ ໂດຍທີ່ຂໍ້ມູນຍັງມີຢູ່ແລ້ວເປັນເລື່ອງທີ່ຍາກຫຼາຍ ຫຼື ເຮັດບໍ່ໄດ້ເລີຍ ວິທີການຄືອາດຈະຕ້ອງແຍກອອກເປັນຕາຕະລາງໃໝ່ ຊຶ່ງເປັນວິທີແກ້ບັນຫາຊົ່ວຄາວເທົ່ານັ້ນ.

ລະບົບຖານຂໍ້ມູນແບບ NoSQL ເຮົາບໍ່ຈຳເປັນຕ້ອງມີ Schema ທີ່ຕາຍຕົວ ຫຼື ບໍ່ຕ້ອງມີ Schema ກ່ອນທີ່ຈະຈັດເກັບຂໍ້ມູນ ຂໍ້ມູນແຕ່ລະແຖວສາມາດຈັດເກັບໄດ້ຕາມຕ້ອງການ ຈະເພີ່ມ ຫຼື ຫຼຸດ ກໍບໍ່ມີ ບັນຫາກັບລະບົບ ເຮັດໃຫ້ເຮົາສາມາດຈັດເກັບຂໍ້ມູນໄດ້ຕາມທີ່ຕ້ອງການປ່ຽນແປງໄດ້ຕະຫຼອດເວລາ ສະ ດວກ ແລະ ວ່ອງໄວ.

#### - **Auto-Sharding**

ເມື່ອຂໍ້ມູນມີຂະໜາດໃຫຍ່ ຫຼື ເຮົາຕ້ອງການເພີ່ມປະສິດທິພາບການ ອ່ານ ແລະ ຂຽນຂໍ້ມູນປະລິ ມານຫຼາຍ ການເຮັດ Sharding ໃນລະບົບ NoSQL Database ຈະກໍ່ການກະຈາຍຂໍ້ມູນໄປຫາ Server ອັດຕະໂນມັດ (Auto-Sharding) ຜູ້ພັດທະນາ (Developer) ບໍ່ຕ້ອງຂຽນໂປຣແກຣມໃນການ ກະຈາຍຂໍ້ມູນເອງຄືກັບ Relational Database.

ການກະຈາຍຂໍ້ມູນອອກໄປຫຼາຍໆ Server ນີ້ຍັງເຮັດໃຫ້ມີຂໍ້ດີຄື ປະຢັດຕົ້ນທຶນໃນການຂະຫຍາຍ ລະບົບ ເພາະເປັນການຂະຫຍາຍແບບແນວນອນ (Scale Out) ຊຶ່ງສາມາດນຳ Server ປົກກະຕິທົ່ວໄປ ມາໃຊ້ງານໄດ້ ບໍ່ຈຳເປັນຕ້ອງເປັນ Enterprise Server

#### - **Replication**

ການສຳເນົາຂໍ້ມູນຈາກເຄື່ອງໜຶ່ງໄປອີກເຄື່ອງໜຶ່ງ (Replication) ເມື່ອ Server ໜຶ່ງເສຍຫາຍ ອີກ ເຄື່ອງໜຶ່ງຈະຂຶ້ນມາເຮັດວຽກແທນທັນທີໂດຍຂໍ້ມູນຂອງແຕ່ລະເຄື່ອງຈະມີຂໍ້ມູນຄືກັນດັ່ງນັ້ນ Replication ເປັນໜຶ່ງຄຸນສົມບັດທີ່ຕອບສະໜອງຕໍ່ການໃຊ້ງານທີ່ຕ້ອງການຄວາມຕໍ່ເນື່ອງໄດ້ຕະຫຼອດເວລາ (High Availability).

#### - **Integrated Caching**

ການຈັດເກັບຂໍ້ມູນທີ່ໃຊ້ງານເລື້ອຍໆ ເຂົ້າໄວ້ໃນ Memory (RAM) ຊຶ່ງເປັນຄຸນສົມບັດເດັ່ນຂອງ NoSQL ທີ່ທັງຫມົດ Caching ໄວ້ໃນຕົວເອງຢູ່ແລ້ວ ເຮົາບໍ່ຈຳເປັນຕ້ອງເຮັດ Cache Layer ຄືກັບ Relational Database ອີກຕໍ່ໄປ ທີ່ຕ້ອງເຮັດ Cache Layer ແຍກຕ່າງຫາກ ແລະ ເບິ່ງແຍງຮັກສາແຍກ ອອກໄປຕ່າງຫາກອີກດ້ວຍ.

## **ຂ. ປະເພດຂອງຖານຂໍ້ມູນ NoSQL**

NoSQL ຖືກແບ່ງປະເພດຕາມລັກສະນະການຈັດເກັບຂໍ້ມູນທີ່ແຕກຕ່າງກັນ ດັ່ງນັ້ນ ການຈະເລືອກ NoSQL Database ໂຕໃດໂຕໜຶ່ງຈະຕ້ອງເບິ່ງອີກວ່າການຈັດເກັບຂໍ້ມູນຂອງຖານຂໍ້ມູນເປັນແບບໃດ 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

ໂດຍສ່ວນໃຫຍ່ແລ້ວ NoSQL ຈະເປັນລິຂະສິດແບບ Open source ຊຶ່ງບໍ່ຕ້ອງເສຍຄ່າໃຊ້ຈ່າຍໃນການນຳມາໃຊ້ງານ ດັ່ງນັ້ນ ເຮົາສາມາດນຳ NoSQL Database ແຕ່ລະຕົວມາຕິດຕັ້ງໃຊ້ງານໄດ້ໂດຍບໍ່ເສຍຄ່າໃຊ້ຈ່າຍໃດໆ (ຟຣີ).

### ງ. ນຳ NoSQL ມາໃຊ້ງານຂະໜາດນ້ອຍໄດ້ ຫຼື ບໍ່?

ຈາກທີ່ກ່າວມາແລ້ວ ຄົງພໍຈະຕອບຄຳຖາມນີ້ໄດ້ວ່າການນຳ NoSQL Database ເມື່ອນຳມາໃຊ້ໃນງານຂະໜາດໃຫຍ່ນັ້ນເພາະສົມຢ່າງແນ່ນອນ ແຕ່ຖ້າເປັນລະບົບທີ່ວຽກໄປຄວນຈະນຳ NoSQL ມາໃຊ້ງານ ຫຼື ບໍ່

ຄຳຕອບຄື ຂຶ້ນຢູ່ກັບລັກສະນະວຽກວ່າເຮົາຈະໃຊ້ຄຸນສົມບັດຫຍັງຂອງ NoSQL ຖ້າເຮົາຕ້ອງການຈັດເກັບຂໍ້ມູນທີ່ບໍ່ຕ້ອງຢຶດຕິດກັບໂຄງສ້າງ (Dynamic Schema) ແລະ ຕ້ອງການເຂົ້າໃຊ້ງານລະບົບທີ່ວ່ອງໄວ (Integrated Caching) ຂໍ້ມູນອາດຈະຢັ້ງຢືນຫຼາຍເທົ່າໃດອາດຈະໃຊ້ NoSQL ໄດ້ຢ່າງແນ່ນອນ

ແຕ່ຖ້າບໍ່ຕ້ອງການໃຊ້ງານ (Dynamic Schema) ບໍ່ຕ້ອງການເຂົ້າໃຊ້ງານທີ່ວ່ອງໄວ (Integrated Caching) ເພາະໃຊ້ Relational Database ກໍເຮັດໄດ້ດີຢູ່ແລ້ວ Database Server ກັບ Web Server ກໍຢູ່ທີ່ Server ດຽວກັນ ຂໍ້ມູນບໍ່ຫລາຍນັ້ນ ບໍ່ຕ້ອງການຈັດເກັບຂໍ້ມູນທີ່ເພີ່ມຂະຫຍາຍຂຶ້ນທຸກມື້ ຜູ້ເຂົ້າໃຊ້ງານກໍບໍ່ໄດ້ຫລາຍ ເບິ່ງແລ້ວວ່າລະບົບບໍ່ມີແນວໂນ້ມຈະຕ້ອງຂະຫຍາຍລະບົບໃນອະນາຄົດອັນໃກ້ ເຈົ້າສາມາດໃຊ້ງານ Relational Database ໄດ້ດີຢູ່ແລ້ວຢ່າງບໍ່ມີບັນຫາ.

### 2.1.6 ຄວາມຮູ້ພື້ນຖານກ່ຽວກັບພາສາ JavaScript

ພາສາ JavaScript ຫຼື ຫຍໍ້ JS ເປັນພາສາຂຽນໂປຣແກຣມທີ່ຖືກພັດທະນາ ແລະ ປະຕິບັດຕາມຂໍ້ກຳນົດມາດຕະຖານຂອງ ECMAScript, ພາສາ JavaScript ນັ້ນເປັນພາສາລະດັບສູງ ຄອມພາຍໃນຂະນະທີ່ໂປຣແກຣມຣັນ (JIT) ແລະ ເປັນພາສາຂຽນໂປຣແກຣມແບບຫຼາຍຂະບວນເຊັ່ນ: ການຂຽນໂປຣແກຣມແບບຂັ້ນຕອນ, ການຂຽນໂປຣແກຣມແບບວັດຖຸ, ພາສາ JavaScript ມີໄວຍະກອນທີ່ຄືກັບພາສາ C ໃຊ້ວົງເລັບເພື່ອກຳນົດບ່ອກຂອງຄຳສັ່ງ ນອກຈາກນີ້ JavaScript ຍັງເປັນພາສາທີ່ມີປະເພດຂໍ້ມູນແບບໄດນາມິກ (Dynamic) ເປັນພາສາແບບ Prototype-based ແລະ First-class function.

ພາສາ JavaScript ນັ້ນຖືວ່າເປັນເທັກໂນໂລຊີຫຼັກຂອງການພັດທະນາເວັບໄຊ (World Wide Web) ມັນເຮັດໃຫ້ໜ້າເວັບສາມາດຕອບໂຕ້ກັບຜູ້ໃຊ້ໄດ້ໂດຍທີ່ບໍ່ຈຳເປັນຕ້ອງລິເຟດໜ້າໃໝ່ (Dynamic website) ເວັບໄຊຈຳນວນຫຼາຍໃຊ້ພາສາ JavaScript ສຳລັບຄວບຄຸມການເຮັດວຽກງານທີ່ດ້ານ Client-side ນັ້ນເຮັດໃຫ້ເວັບບຣາວເຊີຕ່າງໆ ມີ JavaScript engine ທີ່ໃຊ້ສຳລັບປະມວນຜົນສະຄິບຂອງພາສາ JavaScript ທີ່ຮັບເທິງເວັບບຣາວເຊີເນື່ອງຈາກພາສາ JavaScript ເປັນພາສາ

ຂຽນໂປຣແກຣມແບບຫຼາຍຮູບແບບເຮັດໃຫ້ມັນຮອງຮັບການຂຽນໂປຣແກຣມທັງແບບEvent-driven, Functional ແລະ ແບບລຳດັບຂັ້ນຕອນມັນມີ Library (APIs) ສຳລັບເຮັດວຽກກັບຂໍ້ຄວາມ, ວັນທີ, Regular expression ແລະ ໂຄງສ້າງຂໍ້ມູນພື້ນຖານຢ່າງ Array ແລະ Map ຫຼື ທັງ Document Object Model (DOM) ຊຶ່ງເປັນ API ທີ່ໂດຍທົ່ວໄປແລ້ວສາມາດໄດ້ເທິງເວັບບຣາວເຊີ.

ຢ່າງໃດກໍຕາມຕົວຂອງພາສາ JavaScript ເອງບໍ່ໄດ້ມີພັງຊັນສຳລັບອິນພຸດ/ເອົາພຸດ(I/O) ທີ່ມາກັບພາສາເຊັ່ນ: ພັງຊັນກ່ຽວກັບ Network ວຽກກ່ຽວກັບໄຟລ ຫຼື Library ກ່ຽວກັບກຣາບຟິກໂດຍທົ່ວໄປແລ້ວສິ່ງເຫຼົ່ານີ້ຈະຖືກໃຫ້ມາໂດຍ Host environment (ສະພາບແວດລ້ອມທີ່ໃຊ້ຮັບພາສາ JavaScript) ເຊັ່ນ ເວັບເວັບບຣາວເຊີ ຫຼື Node.js ຊຶ່ງຈະແຕກຕ່າງກັນອອກໄປ ຕົວຢ່າງ ເຊັ່ນ:ການຮັບຄ່າໃນເວັບເວັບບຣາວເຊີຈະຜ່ານພັງຊັນ prompt ຊຶ່ງເປັນສ່ວນໜຶ່ງຂອງ Browser Object Model (BOM) ຫຼື ຮັບຄ່າຈາກ HTML ຟອມຊຶ່ງເປັນສ່ວນໜຶ່ງຂອງ Document Object Model (DOM) ໃນຂະນະທີ່ເທິງ Node.js ເຮົາສາມາດຮັບຄ່າໄດ້ຈາກ Input/Output Stream ຂອງ Command line ເຖິງແມ່ນວ່າມັນຈະມີຄວາມຄ້າຍຄືກັນລະຫວ່າງພາສາ Java ແລະ JavaScript ເຊັ່ນ: ຊື່ຂອງພາສາໄວຍະກອນ ຫຼື Library ມາດຕະຖານຕ່າງໆ ຢ່າງໃດກໍຕາມທັງສອງພາສາແຕກຕ່າງກັນຢ່າງສິ້ນເຊີງໃນແງ່ຂອງການອອກແບບພາສາ Java ເປັນພາສາທີ່ມີປະເພດຂໍ້ມູນແບບຄົງທີ່ (Static-typing) ໃນຂະນະທີ່ພາສາ JavaScript ມີປະເພດຂໍ້ມູນແບບໄດນາມິກ (Dynamic-typing) ພາສາ Java ຖືກຄອມພາຍເປັນ Byte-code ກ່ອນທີ່ຈະລົ້ນໃນຂະນະທີ່ພາສາ JavaScript ຈະຄອມພາຍໃນຕອນທີ່ໂປຣແກຣມຮັບພາສາ Java ເປັນພາສາແບບ Class-based ໃນຂະນະທີ່ພາສາ JavaScript ເປັນພາສາແບບ Prototypebased.



## ກ. ປະຫວັດຄວາມເປັນມາຂອງພາສາ JavaScript

ພາສາ JavaScript ຖືກອອກແບບແລະສ້າງໂດຍ Brendan Eich ສໍາລັບເປັນພາສາສະຄິບທີ່ເຮັດວຽກເທິງເວັບບຣາວເຊີ Navigator ທີ່ເປັນຜະລິດຕະພັນຂອງບໍລິສັດ Netscape ເພື່ອເຮັດໃຫ້ໜ້າເວັບທີ່ໃນຕອນທຳອິດນັ້ນເປັນແບບ Static ສາມາດຕອບໂຕ້ກັບຜູ້ໃຊ້ໄດ້ໂດຍທີ່ບໍ່ຈຳເປັນຕ້ອງຮູ້ເພດໜ້າໃໝ່ (Dynamic) ເຊັ່ນ: ການສົ່ງຂໍ້ມູນເບື້ອງໜ້າໄປຍັງ Server ແລະ ລໍຖ້າຮັບຜົນຕອບກັບມາດ້ວຍ AJAX; ພາສາ JavaScript ໄດ້ຖືກເປີດຕົວ ແລະ ເປັນສ່ວນໜຶ່ງຂອງເວັບບຣາວເຊີ Navigator ໃນເດືອນກັນຍາ 1995 ໂດຍໃຊ້ຊື່ວ່າ LiveScript ແລະ ໄດ້ປ່ຽນເປັນ JavaScript ໃນອີກສາມເດືອນຕໍ່ມາ.

ໃນເດືອນພະຈິກ 1996 Netscape ໄດ້ສົ່ງພາສາ JavaScript ໄປຍັງ ECMA International ເພື່ອເປັນຈຸດເລີ່ມຕົ້ນສໍາລັບກຳນົດມາດຕະຖານໃຫ້ທຸກເວັບບຣາວເຊີປະຕິບັດຕາມມາດຕະຖານດັ່ງກ່າວ ເພື່ອໃຫ້ການພັດທະນາ JavaScript engine ເປັນໄປໃນທິດທາງດຽວກັນນັ້ນໃຫ້ເກີດການເປີດຕົວຢ່າງເປັນທາງການສໍາລັບຂໍ້ກຳນົດມາດຕະຖານ ECMAScript ໃນເດືອນມິຖຸນາ 1997 ໃນຊ່ວງເວລາຫຼັງຈາກນີ້ ບໍລິສັດຕ່າງໆ ທີ່ພັດທະນາເວັບບຣາວເຊີຕ່າງກໍຍັງພັດທະນາ JavaScript engine ບໍ່ເປັນໄປໃນທິດທາງດຽວກັນເທົ່າໃດ ນັ້ນເຮັດໃຫ້ນັກພັດທະນາເວັບຕ້ອງຂຽນໂຄດຫຼາຍເວັບເພື່ອໃຫ້ເຮັດວຽກໄດ້ໃນທຸກເວັບບຣາວເຊີຈົນກະທັ້ງໃນເດືອນກໍລະກົດ 2008 ໄດ້ມີການຈັດການປະຊຸມຂຶ້ນທີ່ Oslo ຈາກອົງກອນ ແລະ ຝ່າຍຕ່າງໆ ທີ່ພັດທະນາ JavaScript engine ເຮັດໃຫ້ເກີດຂໍ້ຕົກລົງຂຶ້ນໃນຕົ້ນປີ 2009 ເພື່ອລວບລວມງານທີ່ກ່ຽວຂ້ອງທັງໝົດຂອງພາສາ JavaScript ແລະ ຊຸກຍູ້ພາສາໃຫ້ຢ່າງໄປຂ້າງໜ້າ ນັ້ນເຮັດໃຫ້ເກີດຂໍ້ກຳນົດມາດຕະຖານ ECMAScript ເວີຊັນທີ 5 (ES5) ອອກມາໃນເດືອນທັນວາ 2009 ແລະ ກ່ອນໜ້ານີ້ໃນປີ 2008 Google ໄດ້ເປີດຕົວເວັບ

ບຣາວເຊີ Chrome ທີ່ມາພ້ອມກັບ V8 JavaScript engine ທີ່ມີແນວຄິດໃນການພັດທະນາແບບຄອມພາຍໃນຕອນທີ່ໂປຣແກຣມຮັ້ນ (Just-in-time compilation: JIT) ຊຶ່ງມັນເຮັດວຽກໄດ້ໄວກວ່າເຮັດໃຫ້ຜູ້ພັດທະນາເວັບບຣາວເຊີອື່ນໆ ຕ້ອງປັບປຸງ JavaScript engine ຂອງພວກເຂົາໃຫ້ເຮັດວຽກໃນຮູບແບບ JIT ຫຼັງຈາກທີ່ພັດທະນາຕໍ່ເນື່ອງມາອີກຫຼາຍປີ ໃນປີ 2015 ໄດ້ມີການເພີ່ມຄຸນສົມບັດໃໝ່ໆ ທີ່ຫຼາກຫຼາຍເຂົ້າມາ ຊຶ່ງຖືວ່າເປັນການປ່ຽນແປງເທື່ອສໍາຄັນ ແລະ ເຮັດໃຫ້ເກີດຂໍ້ກຳນົດມາດຕະຖານ ECMAScript 2015 ຫຼື ເວີຊັນທີ 6 (ES6) ຈົນເຖິງໃນປີ 2015 ຕອນນີ້ເບິ່ງຄືວ່າພາສາ JavaScript ຈະພັດທະນາມາຈົນເຖິງທີ່ສຸດແລ້ວ ເຮັດໃຫ້ລະຫວ່າງປີ 2016 - 2019 ເວີ

ຊັ້ນໃໝ່ຂອງ ECMAScript ທີ່ຖືກເຜີຍແຜ່ອອກມາໃນແຕ່ລະປີມີການປ່ຽນແປງແລະເພີ່ມຄຸນສົມບັດ ພຽງນ້ອຍໆເທົ່ານັ້ນ.

## ຂ. ຄຸນສົມບັດຂອງພາສາ JavaScript

ECMAScript 2015 (ES6) ເປັນພາສາ JavaScript ທີ່ຖືວ່າພັດທະນາມາຈົນເຖິງຈຸດສູງສຸດ ແລ້ວກໍ່ວ່າໄດ້ ມັນຖືກເຜີຍແຜ່ໃນເດືອນມິຖຸນາ 2015 ຊຶ່ງໃນເວີຊັນນີ້ ໄດ້ເພີ່ມໄວຍະກອນໃໝ່ຂອງ ພາສາຫຼວງຫລາຍເຊັ່ນ: ການສ້າງຄາດດ້ວຍຄໍາສັ່ງ class ການສ້າງໂມດູນ ແລະ ໃຊ້ງານມັນດ້ວຍ ຄໍາສັ່ງ import ແລະ export ແລະ ຄໍາສັ່ງສໍາລັບປະກາດຕົວປ່ຽນ let ແລະ ປະກາດຄ່າຄົງທີ່ const ຊຶ່ງເຮັດໃຫ້ຕົວປ່ຽນສາມາດມີຂອບເຂດໃນບ່ອນທີ່ມັນຖືກສ້າງຂຶ້ນໄດ້ ແລະ ສິ່ງອື່ນໆທີ່ຖືກເພີ່ມ ເຂົ້າມາເປັນຈຳນວນຫລາຍເຊັ່ນ: Map, Set, WeakMap, Promise, Reflection, Proxies, Template string ແລະ ອື່ນໆ.

ໃນເດືອນມິຖຸນາ 2016 ໄດ້ມີການເປີດໂຕເວີຊັນ 7 ຫຼື ECMAScript 2016 (ES7) ໄດ້ມີການ ເພີ່ມຕົວດໍາເນີນການຍົກກຳລັງ (ທີ່ກ່ອນໜ້ານີ້ເຮົາຈະໃຊ້ຜ່ານຟັງຊັນ Math.pow) ຄໍາສັ່ງ await async ສໍາລັບການຂຽນໂປຣແກຣມທີ່ເຮັດວຽກບໍ່ພ້ອມກັນ ແລະ ຟັງຊັນ includes ຂອງອາ ເລ ແລະ ໃນປະຈຸບັນພາສາ JavaScript ຖືກພັດທະນາມາຈົນເຖິງ ECMAScript 2020 (ES11) ຊຶ່ງມີການປ່ຽນແປງທີ່ເພີ່ມຂຶ້ນບໍ່ເທົ່າໃດຫຼັງຈາກ ES7.

## ຄ. JavaScript engine ແມ່ນຫຍັງ?

JavaScript engine ຄືໂປຣແກຣມຄອມພິວເຕີທີ່ໃຊ້ສໍາລັບປະມວນຜົນໂຄດຂອງພາສາ JavaScript ຊຶ່ງ JavaScript engine ໃນຊ່ວງເລີ່ມຕົ້ນເປັນພຽງແຕ່ຕົວປ່ຽນພາສາ (Interpreter) ເທົ່າ ນັ້ນ ແຕ່ໃນປະຈຸບັນໄດ້ມີການພັດທະນາມາໃຫ້ຢູ່ໃນຮູບແບບຂອງຄອມພາຍເລີທີ່ມີການຄອມພາຍໃນ ຕອນທີ່ໂປຣແກຣມລັ້ນ (Just-in-time compilation: JIT) ເພື່ອເພີ່ມປະສິດທິພາບການເຮັດວຽກງານ ຂອງໂປຣແກຣມ ໂດຍທົ່ວໄປແລ້ວ JavaScript engine ຈະຖືກພັດທະນາໂດຍຜູ້ພັດທະນາ ເວັບບຣາວເຊີທີ່ປະຕິບັດຕາມຂໍ້ກຳນົດມາດຕະຖານຂອງ ECMAScript.

### 2.1.7 ຄວາມຮູ້ພື້ນຖານກ່ຽວກັບ Application Programming Interface (API)

ຫຼາຍຄົນອາດຈະສົງໄສວ່າ API ຄືຫຍັງ ຄວາມຈິງແລ້ວຫຍໍ້ມາຈາກ Application Program Interface (API) ຊຶ່ງຄື ຄໍາສັ່ງ (Code) ທີ່ອະນຸຍາດໃຫ້ software program ສາມາດສື່ສານລະຫວ່າງ ກັນໄດ້ ຖ້າຈະເວົ້າໃນພາສາຄົນຂຽນ program ແລ້ວ API ເປັນຊ່ອງທາງສໍາລັບຂໍໃຊ້ບໍລິການຄໍາສັ່ງ

ຈາກ operation system (OS) ຫຼື application ອື່ນໆ ຊຶ່ງມັນໃຊ້ງານໂດຍຕິດຕັ້ງ function ແລະ ເອີ້ນໃຊ້ງານຕາມ document ທີ່ຂຽນໄວ້.

### ສ່ວນປະກອບຂອງ APIs

APIs ສ້າງຂຶ້ນຈາກສ່ວນສໍາຄັນ 2 ຢ່າງຄື:

- ຂໍ້ກຳນົດທີ່ຈະອະທິບາຍການແລກປ່ຽນຂໍ້ມູນລະຫວ່າງ program ຊຶ່ງເຮັດອອກມາໃນລັກສະນະ document ເພື່ອບອກວ່າ request/response ຕ້ອງເປັນຢ່າງໃດ.
- Software ທີ່ຂຽນຂຶ້ນຕາມຂໍ້ກຳນົດ ແລະ ກໍານົດເສຍແຕ່ອອກໄປໃຫ້ໃຊ້ງານ.

ໂດຍປົກກະຕິແລ້ວ Application ທີ່ມີ APIs ຈະຕ້ອງຖືກຂຽນເປັນພາສາ Programming ແລະ ພັດທະນາເພີ່ມໄດ້ງ່າຍ ຈຶ່ງຈຳເປັນຕ້ອງມີການກວດສອບໂຄງສ້າງ API ສະນັ້ນ API ທີ່ດີ ຜູ້ທີ່ອອກແບບຕ້ອງໃຫ້ຄວາມສໍາຄັນໃນການ test ເພື່ອກວດສອບ logic ທີ່ສາມາດເກີດຂຶ້ນໄດ້ຈາກການໃຊ້ງານ.

### ➤ ການໃຊ້ງານ APIs

ປັດຈຸບັນນີ້ API ຖືກໃຊ້ງານໃນ application ເພື່ອສື່ສານກັບ user ໂດຍບໍ່ຈຳເປັນຕ້ອງມີຄວາມຮູ້ ບໍລິສັດໃຫຍ່ໆຫຼາຍບໍລິສັດມີການເປີດ API ໃຫ້ພາຍນອກເຂົ້າມາໃຊ້ງານ ເຊັ່ນ facebook, google, twitter ຜູ້ພັດທະນາລະບົບທີ່ສົນໃຈ ສາມາດນຳເອົາ API ເຫຼົ່ານີ້ໄປໄປຕໍ່ຍອດ ຊຶ່ງທາງບໍລິສັດກໍສາມາດຂະຫຍາຍຖານລູກຄ້າອອກໄປໄດ້ອີກ ຮູບແບບການນຳເອົາ API ໄປໃຊ້ງານມີດັ່ງນີ້:

## 1. Libraries and frameworks

API ມັກຈະເອົາໄປໃຊ້ເປັນ software library ຊຶ່ງຂຽນຂຶ້ນຕາມ document ໃນຮູບແບບພາສາ program ທີ່ຕ່າງກັນອອກໄປ ຕາມຄວາມເໝາະສົມກັບວຽກເພື່ອເອົາໄປເຮັດເປັນ framework ໃຫ້ກັບລະບົບໃຊ້ໃນການສື່ສານຫາກັນ.

## 2. Operating Systems

API ສາມາດໃຊ້ງານໃນການສື່ສານລະຫວ່າງ application ແລະ operating system ເຊັ່ນ: POSIX ຫຼື ມາດຕະຖານການສື່ສານຂອງ OS ເອງກໍມີ API ເປັນ command line ເພື່ອຄວບຄຸມການເຮັດວຽກຂອງ OS.

## 3. Remote APIs

Remote APIs ເຮັດໃຫ້ developer ສາມາດເຂົ້າຄວບຄຸມຊັບພະຍາກອນຜ່ານທາງ protocol ເພື່ອໃຫ້ມີມາດຕາຖານການສື່ສານດຽວກັນ ເຖິງແມ່ນວ່າຈະເປັນຄົນລະ technology ເຊັ່ນ Database

API ສາມາດອະນຸຍາດໃຫ້ developer ເຂົ້າມາດົງຂໍ້ມູນໃນ database ຫຼາກຫຼາຍຊະນິດໄດ້ ຜ່ານ function ດຽວກັນ ສະນັ້ນ remote API ຈຶ່ງຖືກໃຊ້ເລື້ອຍໃນ maintenance ດ້ວຍການເຮັດວຽກທີ່ຝັງ client ໃຫ້ໄປດົງຂໍ້ມູນຈາກ server ກັບລົງມາເຮັດວຽກ.

#### 4. Web APIs

ນິຍົມໃຊ້ກັນຫລາຍໃນປະຈຸບັນ ເພາະຢູ່ໃນກຸ່ມຂອງ HTTP ແລະ ຂະຫຍາຍອອກໄປສູ່ຮູບແບບ XML ແລະ JSON ຊຶ່ງໂດຍລວມແລ້ວກໍຄືຢູ່ເທິງ web service ເຊັ່ນ:

- SOAP (Simple Object Access Protocol) ໃຊ້ XML format ສົ່ງຂໍ້ມູນ.
- REST (Representational State Transfer) ສາມາດໃຊ້ XML ຫຼື JSON format ສົ່ງຂໍ້ມູນ.

#### ຕົວຢ່າງ API ທີ່ນິຍົມໃນປະຈຸບັນ

- Google Maps API: ເປີດໃຫ້ໃຊ້ງານເພື່ອນຳເອົາແຜນທີ່ຂອງ Google ມາລົງໃນ webpage ໂດຍອາໄສ JavaScript ຫຼື Flash.
- YouTube APIs: Google ຍອມໃຫ້ developer ສາມາດນຳເອົາ Clip video ເທິງ YouTube ໄປລົງໃນ website ຫຼື application ໄດ້.
- Flickr API: ເພື່ອໃຫ້ developer ສາມາດເຂົ້າເຖິງ ຄັງຮູບພາບໃນ community
- Twitter APIs: ມີ REST API ໃຫ້ຄົ້ນຫາແລ້ວກວດສອບຂໍ້ມູນ trends ໄດ້.
- Amazon Product Advertising API: ເປີດ API ໃຫ້ໃຊ້ຄົ້ນຫາສິນຄ້າ ແລະ ການໂຄສະນາ ຜ່ານທາງ website.

#### 2.1.8 ຄວາມຮູ້ພື້ນຖານກ່ຽວກັບ Nodejs

Node.js ເປັນ open-source ແລະ cross-platform JavaScript runtime environment ທີ່ກຳລັງໄດ້ຮັບຄວາມນິຍົມສູງ ໂດຍທົ່ວໄປເຮົາຈະໃຊ້ JavaScript ໃນຝັ່ງ client ແຕ່ Node.js ເຮັດໃຫ້ເຮົາໃຊ້ JavaScript ໃນຝັ່ງ Server ໄດ້ດ້ວຍຊຶ່ງ Node.js ສາມາດ run ໄດ້ເທິງ platform ທີ່ຫຼາກຫຼາຍທັງ Windows, Linux, Unix, Mac OS X ແລະ ອື່ນໆ.



## ຮູບທີ 12 Nodejs

ຈຸດເດັ່ນທີ່ສຸດຂອງ Node.js ຄືມັນເຮັດວຽກແບບ asynchronous ຜູ້ອ່ານອາດຈະສົງໃສວ່າແລ້ວມັນດີແນວໃດ? ລອງມາເບິ່ງຕົວຢ່າງການຈັດການໄຟລເມື່ອມີການຮ້ອງຂໍຈາກ client ມາທີ່ server ຂອງ PHP ກັບ Node.js ທຽບກັນເບິ່ງເພື່ອໃຫ້ເຂົ້າໃຈຫຼາຍຂຶ້ນ.

- ກໍລະນີ PHP ເມື່ອມີການຮ້ອງຂໍເຂົ້າມາມັນຈະເຮັດດັ່ງນີ້:
  1. ສົ່ງ task ໄປທີ່ລະບົບໄຟຂອງຄອມພິວເຕີ.
  2. ລໍຖ້າຈົນກະທັ້ງລະບົບໄຟລເປີດ ແລະ ອ່ານໄຟລສໍາເລັດ.
  3. ສົ່ງເນື້ອຫາຂອງໄຟລກັບມາໃຫ້ client.
  4. ພ້ອມສໍາລັບຮັບຄໍາຮ້ອງຂໍຖັດໄປ.
- ກໍລະນີ Node.js ເມື່ອມີການຮ້ອງຂໍເຂົ້າມາມັນຈະເຮັດດັ່ງນີ້:
  1. ສົ່ງ task ໄປທີ່ລະບົບໄຟຂອງຄອມພິວເຕີ.
  2. ພ້ອມສໍາລັບຮັບຄໍາຮ້ອງຂໍຖັດໄປ.
  3. ພໍລະບົບໄຟລເປີດ ແລະ ອ່ານໄຟລສໍາເລັດ server ຈະສົ່ງເນື້ອຫາຂອງໄຟລກັບມາໃຫ້ client.

ຈາກຕົວຢ່າງຂ້າງເທິງຈະເຫັນວ່າ Node.js ຈະຕັດຂັ້ນຕອນການລໍຖ້າຖິ້ມແລ້ວໄປເຮັດຄໍາຮ້ອງຖັດໄປເລີຍທີ່ ເປັນແບບນີ້ເພາະ Node.js ຈະ run ແບບ single-threaded ແລະ ໃນ Library ມາດຕະຖານກໍຈະມີເຊັດຂອງ asynchronous I/O primitives ທີ່ຊ່ວຍປ້ອງກັນໂຄດ JavaScript ຈາກການ blocking ເຮັດໃຫ້ລະບົບຄ່ອງແຄ້ວ ແລະ ມີປະສິດທິພາບຫຼາຍຂຶ້ນ.

Synchronous vs Asynchronous	
<p>Synchronous ຄືການ run ໂຄດຕາມລຳດັບທີ່ເຮົາຂຽນໄວ້ ເຊັ່ນ</p> <pre>alert(1); alert(2); alert(3);</pre> <p>ຜົນລັບທີ່ໄດ້ຄືໜ້າຈໍຈະສະແດງຜົນ 1 ຈາກນັ້ນຈຶ່ງສະແດງຜົນ 2 ແລ້ວສຸດທ້າຍຈຶ່ງສະແດງຜົນ 3 ຕາມລຳດັບ</p>	<p>Asynchronous ຄືການຮັບໂຄດທີ່ບໍ່ຈຳເປັນຕ້ອງເປັນໄປຕາມລຳດັບທີ່ເຮົາຂຽນໄວ້ ເຊັ່ນ:</p> <pre>alert(1); setTimeout(() =&gt; alert(2), 0); alert(3);</pre> <p>ໃນຕົວຢ່າງນີ້ໂຄດ alert(2) ໃຊ້ເວລາດຳເນີນການດົນກວ່າ ເຮັດໃຫ້ຜົນລັບທີ່ໄດ້ຄືໜ້າຈໍຈະສະແດງຜົນ 1 ຈາກນັ້ນຈຶ່ງສະແດງຜົນ 3 ແລ້ວສຸດທ້າຍຈຶ່ງສະແດງຜົນ 2</p>
Blocking vs Non-blocking	
<p>Blocking ໝາຍເຖິງການທີ່ເຮົາບໍ່ສາມາດດຳເນີນການຕໍ່ໄປໄດ້ຈົນກວ່າຕົວດຳເນີນການທີ່ກຳລັງລັນຢູ່ຈະສຳເລັດເສຍກ່ອນ ເຊັ່ນ</p> <pre>alert(1); var value = localStorage.getItem('foo'); alert(2);</pre> <p>ຄຳສັ່ງ localStorage ຈະເປັນຕົວ blocking ເຮັດໃຫ້ບໍ່ສາມາດເຮັດຄຳສັ່ງ alert(2) ໄດ້ຈົນກວ່າມັນຈະດຳເນີນການສຳເລັດ ດັ່ງນັ້ນຜົນລັບທີ່ໄດ້ຄືໜ້າຈໍຈະສະແດງຜົນ 1 ຈາກນັ້ນຈຶ່ງລໍຖ້າຈົນກວ່າຄຳສັ່ງ localStorage ຈະສຳເລັດຈຶ່ງສະແດງຜົນ 2</p>	<p>Non-blocking ໝາຍເຖິງການໂຕທີ່ດຳເນີນການສາມາດເຮັດຄຳສັ່ງຖັດໄປໄດ້ເລີຍໂດຍບໍ່ຕ້ອງລໍຖ້າໃຫ້ຄຳສັ່ງເດີມເຮັດສຳເລັດກ່ອນເຊັ່ນ:</p> <pre>alert(1); fetch('example.com').then(() =&gt; alert(2)); alert(3);</pre> <p>ໃນຕົວຢ່າງນີ້ ຄຳສັ່ງ fetch ເປັນ non-blocking operation ດັ່ງນັ້ນຜົນລັບທີ່ໄດ້ຄືໜ້າຈໍຈະສະແດງຜົນ 1 ຈາກນັ້ນຈຶ່ງສະແດງຜົນ 3 ແລ້ວພໍຄຳສັ່ງ fetch ສຳເລັດກໍສະແດງຜົນ 2</p>

## ➤ ປະຫວັດ Node.js

ແຕ່ເດີມພາສາ JavaScript ຖືກພັດທະນາມາເພື່ອໃຊ້ສຳລັບ Browser ທີ່ຊື່ Netscape Navigator ໃນປີ 1995 ໃນຕອນນັ້ນ Netscape ຕັ້ງໃຈຈະຂາຍ Web Server ທີ່ມີ environment ຊື່ Netscape LiveWire ຊຶ່ງສາມາດສ້າງ dynamic page ໂດຍໃຊ້ JavaScript ທາງຝັ່ງ server ອີກດ້ວຍ ແຕ່ໜ້າເສຍດາຍທີ່ Netscape LiveWire ບໍ່ປະສົບຄວາມສຳເລັດ ແລະ ການໃຊ້ JavaScript ທາງຝັ່ງ Server ກໍບໍ່ໄດ້ຮັບຄວາມນິຍົມເລີຍຈົນກະທັ້ງ Node.js ຖືກກຳເນີດຂຶ້ນມາ.

ສິ່ງທີ່ເຮັດໃຫ້ Node.js ເປັນທີ່ນິຍົມຂຶ້ນມາຄືການທີ່ມັນມາໃນຊ່ວງເວລາທີ່ເໝາະສົມ ເມື່ອທຽບກັບ JavaScript ທີ່ເກີດມາຕັ້ງແຕ່ປີ 1995 ແລ້ວ Node.js ຫາກໍເກີດມາເມື່ອປີ 2009 ເທົ່ານັ້ນ ຕ້ອງຂອບໃຈ "Web 2.0" applications ເຊັ່ນ Flickr, Gmail ແລະ ອື່ນໆທີ່ສະແດງໃຫ້ໂລກຮູ້ວ່າເວັບສະ

ໄໝໃໝ່ຄວນໜ້າຕາເປັນຢ່າງໃດ.

ບໍ່ດົນກ່ອນທີ່ Node.js ຈະເກີດ ນັກພັດທະນາເບື້ອງໜຶ່ງ browser ຊຶ່ງທັງຫຼາຍແຂ່ງຂັນກັນເຮັດວຽກຢ່າງໜັກເພື່ອຈະໃຊ້ JavaScript ໃຫ້ໄດ້ດີທີ່ສຸດ ແລະ ຫາທາງເຮັດໃຫ້ JavaScript ສາມາດຮັບໄດ້ໄວຫຼາຍຂຶ້ນເພື່ອໃຫ້ຜູ້ໃຊ້ງານໄດ້ຮັບປະສິດທິພາບທີ່ດີທີ່ສຸດ ຊຶ່ງຜົນຈາກການແຂ່ງຂັນນີ້ເຮັດໃຫ້ເກີດການພັດທະນາ Chrome V8 (open-source JavaScript engine ຂອງ The Chromium Project) ຂຶ້ນມາ ແລະ Node.js ກໍໃຊ້ engine.

ແຕ່ການທີ່ Node.js ເປັນທີ່ນິຍົມຂຶ້ນມາບໍ່ແມ່ນແຕ່ວ່າມັນມາຖືກທີ່ຖືກເວລາເທົ່ານັ້ນ ແຕ່ມັນໄດ້ສະແດງໃຫ້ເຫັນແລ້ວວ່າ ການອອກແບບ ແລະ ແນວຄິດຂອງມັນຊ່ວຍນັກພັດທະນາທັງຫຼາຍໃຫ້ສາມາດໃຊ້ JavaScript ທາງຝັ່ງ server ໄດ້ງ່າຍຂຶ້ນຫຼາຍອີກດ້ວຍ.

### 2.1.9 ຄວາມຮູ້ພື້ນຖານກ່ຽວກັບ Reactjs (JavaScript Library)

React ເປັນເທັກໂນໂລຢີໜຶ່ງ ທີ່ມາແຮງຫຼາຍໂດຍສ້າງມາຈາກພື້ນຖານແນວຄວາມຄິດແບບ MVC (Model View Controller) ຊຶ່ງໝາຍຄວາມວ່າ React ມີໜ້າທີ່ຈັດການກັບ Model ຫຼື View ແຕ່ສ່ວນໃຫຍ່ຈະເປັນ View ກ່ອນໜ້ານັ້ນເວລາຈະຂຽນໜ້າເວັບເຮົາກໍຈະເຮັດຜ່ານ HTML ມີການໃຊ້ CSS ໃນການປັບປຸງໜ້າຕາຂອງ UI ແຕ່ໃນ React ຖ້າຈະສ້າງໜ້າເວັບຂຶ້ນມາໄດ້ນັ້ນ ເຮົາຈະໃຊ້ເປັນ Component ປຸງໄດ້ວ່າ Component ເປັນ Block ສ່ວນຍ່ອຍຂອງເວັບເຮົາທີ່ຈະສ້າງອອກມາ.

#### ສະຫຼຸບງ່າຍໆດັ່ງນີ້

- React ເປັນ Javascript Library ສ້າງ ແລະ ພັດທະນາຂຶ້ນໂດຍ Facebook ຂຽນໄດ້ແຕ່ UI ເທົ່ານັ້ນ ແລະ ເປີດໃຫ້ໃຊ້ຟຣີ.
- React ໃຊ້ໂຄດ HTML , CSS ແລະ Javascript.
- React ມີ 3 ຄອນເຊບທີ່ເຮົາຕ້ອງຮຽນຮູ້ຄື Component State ແລະ Props.

#### ➤ ຂໍ້ດີ ແລະ ຂໍ້ເສຍຂອງ React

##### ກ. ຂໍ້ດີ

- Component ເຂົ້າໃຈງ່າຍເຮົາສາມາດຮຽນຮູ້ໄດ້ດ້ວຍຕົວເອງ.
- Tool ຫຼາຍພຽງ React ຢ່າງດຽວກໍສາມາດຂຽນເວັບໄດ້ທັງເວັບແລ້ວ ໂດຍບໍ່ຕ້ອງຫາ Tool ເພີ່ມເຕີມ ແລະ ຍັງມີ Tool ພັດທະນາອອກມາຢູ່ເລື້ອຍໆສາມາດເຮັດ App ໄດ້ React ມີເຄື່ອງມືໜຶ່ງທີ່ຊື່ວ່າ React Native ເປັນການຂຽນ JavaScript ແລ້ວແປງເປັນ App ແບບ Native ໄດ້ທັງເທິງ Android ແລະ Ios.

## ຂ. ຂໍ້ເສຍ

- ຕ້ອງມີພື້ນຖານໃນ Javascript ໃນລະດັບໜຶ່ງ ຄົນທີ່ສົນໃຈຮຽນກໍຈະລຳບາກໜ້ອຍໜຶ່ງຖ້າບໍ່ໄດ້ມີພື້ນຖານ Javascript ອາດຈະຕ້ອງໃຊ້ເວລາໜ້ອຍໜຶ່ງ.
- Documentation ອ່ານຍາກ React ມີ Documentation ທີ່ຍັງບໍ່ຄ່ອຍດີໃນອະນາຄົດອາດຈະມີການພັດທະນາອີກ.

### 2.1.10 ຄວາມຮູ້ພື້ນຖານກ່ຽວກັບ GraphQL (GraphQL API)

GraphQL ຄືພາສາສຳລັບການເຂົ້າເຖິງຂໍ້ມູນ (Query Language) ເພື່ອການໃຊ້ງານ API ຂອງລະບົບ ແລະ ຈະປະມວນຜົນຄຳສັ່ງທີ່ຝັງ server ຫຼື ທີ່ເອີ້ນວ່າ server-side runtime ໂດຍໃຊ້ໂຄງສ້າງຂໍ້ມູນທີ່ເຮົາກຳນົດໄວ້. ແຕ່ GraphQL ບໍ່ໄດ້ພັດທະນາຂຶ້ນມາເພື່ອແທນທີ່ພາສາສຳລັບການເຂົ້າເຖິງຂໍ້ມູນເຊັ່ນ SQL ຫຼື ເຮັດໜ້າທີ່ເປັນລະບົບຈັດເກັບຖານຂໍ້ມູນ (storage engine) ແຕ່ຢ່າງໃດ.

ໃນປະຈຸບັນວິທີທີ່ເຂົ້າເຖິງຂໍ້ມູນຜ່ານເວັບເຊີວິດທີ່ໄດ້ຮັບຄວາມນິຍົມກໍຈະເປັນ REST API ຜ່ານທາງ HTTP Methods ຕົວຢ່າງເຊັ່ນ: ຕ້ອງການດຶງຂໍ້ມູນຜູ້ໃຊ້ງານທັງໝົດ ກໍສາມາດຮຽກຜ່ານ API ໄດ້ດັ່ງນີ້:

GET /users

ຫຼື ຫາກຕ້ອງການດຶງເພື່ອນທັງໝົດຂອງຜູ້ໃຊ້ງານ ID 25

GET /users/25/friends

ແຕ່ຫາກຕ້ອງການດຶງເບີໂທສັບຂອງເພື່ອນທີ່ເປັນເພື່ອນກັບຜູ້ໃຊ້ງານ ID 25 ຈະຕ້ອງດຶງ ແລະ ຂຽນໂປຣແກຣມແບບໃດ? ໂດຍມີການຈັດເກັບຂໍ້ມູນດັ່ງນີ້:

ຈາກຕົວຢ່າງນີ້ຈະເຫັນໄດ້ວ່າເຮົາຈຳເປັນຕ້ອງມີ API 2 end point ນຳກັນຄື GET /users/:id/friends ເພື່ອດຶງຂໍ້ມູນເພື່ອນຂອງຜູ້ໃຊ້ ID 25 ຈາກນັ້ນ ກໍວິນລຸບດຶງຂໍ້ມູນຜູ້ໃຊ້ໂດຍໃຊ້ GET /users/:id ເພື່ອດຶງຂໍ້ມູນເບີໂທສັບອີກຄັ້ງ ແລະ ວິທີນີ້ກໍຈະໄດ້ຂໍ້ມູນອື່ນໆ ທີ່ບໍ່ຈຳເປັນຂອງຜູ້ໃຊ້ມາອີກ ບໍ່ວ່າຈະເປັນຊື່, ຮູບພາບ ແລະ ອີເມວໃນຄວາມເປັນຈິງແລ້ວ ອົງກອນໃຫຍ່ໆທີ່ມີຫຼາກຫຼາຍພະແນກ ຕ່າງກໍຮ້ອງຂໍຂໍ້ມູນທີ່ບໍ່ຄືກັນເຊັ່ນ ບາງພະແນກຕ້ອງການສະເພາະຂໍ້ມູນພະນັກງານ, ບາງພະແນກຕ້ອງການຂໍ້ມູນພະນັກງານພ້ອມຂໍ້ມູນເງິນເດືອນ ຫຼື ບາງພະແນກຕ້ອງການຂໍ້ມູນພະນັກງານກັບຈຳນວນວັນທີ່ຂາດ ຫຼື ລາເທົ່ານັ້ນເປັນຕົ້ນ ການໃຊ້ງານ REST API ຈຶ່ງສ້າງຄວາມລຳບາກຕໍ່ການພັດທະນາ ແລະ ດູແລຮັກສາເຊີວິດ.



User		Relationship	
Property Name	Type	Property Name	Type
id	Id	id	Id
name	String	user_one_id	Id
image	String	user_two_id	Id
mobile	String		
email	String		

<https://stackdeveloper.co>

ດ້ວຍເທດນີ້ GraphQL ຈຶ່ງໄດ້ເຂົ້າມາເພື່ອແກ້ບັນຫາໃນຈຸດນີ້ໂດຍສະເພາະ ເພື່ອໃຫ້ເຮົາສາມາດດຶງຂໍ້ມູນໄດ້ກົງກັບຄວາມຕ້ອງການ ຫຼຸດຄວາມຊັບຊ້ອນໃນການຂຽນໂຄດເພື່ອດຶງຂໍ້ມູນ ສາມາດຈັດການ ແລະ ດູແລຮັກສາໂຄດໃນຝັ່ງ server-side ໄດ້ງ່າຍຫຼາຍຢ່າງຂຶ້ນ ແລະ ທີ່ສຳຄັນ GraphQL ບໍ່ໄດ້ຜູກຕິດກັບ database ແລະ ທີ່ຈັດເກັບຂໍ້ມູນໃດໆທັງນັ້ນ.

### 2.1.11 ຄວາມຮູ້ພື້ນຖານກ່ຽວກັບ expressjs (Nodejs Library)

Express.js ເປັນ Web Application Framework ຊຶ່ງໄດ້ຮັບຄວາມນິຍົມຫຼາຍສຳລັບເຮັດວຽກເທິງ platform ຂອງ Node.js ຊຶ່ງເປັນ Server ໂຕໜຶ່ງໂດຍທັງ Express.js ແລະ Node.js ຕ່າງກໍໃຊ້ພາສາ JavaScript ໃນການພັດທະນາ ຖ້າເປັນ Web Application Framework ໃນສະໄໝກ່ອນ ຄົນທີ່ພັດທະນາຈະຕ້ອງມີຄວາມຮູ້ຫຼາຍກວ່າ 1 ພາສາ, ພາສາທີ່ເຮັດວຽກທາງຝັ່ງ Server ຢ່າງ PHP ຫຼື ASP ແລະ ພາສາທີ່ເຮັດວຽກທາງຝັ່ງ Client ຢ່າງ JavaScript ເພື່ອຫຼຸດຄວາມຫຍຸ້ງຍາກທັງໝົດເຖິງເວລາໃນການຕ້ອງຮຽນຮູ້ຫຼາຍໆພາສາເຮັດໃຫ້ເກີດ Node.js ກັບ Express.js ພຽງແຕ່ມີຄວາມຮູ້ JavaScript ກໍສາມາດຂຽນໄດ້ທັງ Server ແລະ Client ນອກຈາກນີ້ ຖ້າໃຜເຄີຍຂຽນ JavaScript ຈະຮູ້ວ່າມັນມີການຕອບສະໜອງທີ່ວ່ອງໄວແນ່ນອນວ່າ Express.js ກໍຍົກເອົາມາເປັນຂໍ້ເດັ່ນໃນເລື່ອງຄວາມໄວ ໃນເລື່ອງການຮຽນຮູ້ການຂຽນ Express.js ຈະໃຊ້ຮູບແບບທີ່ງ່າຍໃນການຮຽນຮູ້ຫລາຍທີ່ສຸດສຳລັບການພັດທະນາ Express.js ໃນເວັບໄຊຈະເວົ້າເຖິງການໃຊ້ Routing (ການກຳນົດເສັ້ນທາງຂອງລະບົບ) ແລະ Middleware (ການຮັບສົ່ງຂໍ້ມູນຂອງລະບົບ)

ສາມາດຂຽນໄດ້ໃນຮູບແບບ MVC ສ່ວນການເຊື່ອມຕໍ່ກັບຖານຂໍ້ມູນສາມາດໃຊ້ MongoDB ຫຼື ຈະໃຊ້ MySQL ກໍໄດ້ສໍາລັບນາມສະກຸນຂອງໄຟລຕີ .js ຂະນະນີ້ໄດ້ພັດທະນາມາເຖິງເວີຊັນທີ່ 4.x ແລ້ວ.

### 2.1.12 ຄວາມຮູ້ພື້ນຖານກ່ຽວກັບ Progressive Web App (PWA)

Progressive Web Apps (PWA) ຄືມາດຕະຖານການເຮັດເວັບຈາກ Google ທີ່ນໍາຈຸດເດັ່ນຂອງ Website ແລະ Application ມາລວມກັນ ຜູ້ໃຊ້ສາມາດເຂົ້າເວັບໄຊ ແລະ ໃຊ້ງານເປັນ Application ເລີຍທັງໝ້າຕາ ແລະ ຟີເຈີ ແຕ່ໂດຍເບື້ອງໜຶ່ງແລ້ວຍັງເປັນເທັກໂນໂລຢີຂອງເວັບໄຊຢູ່ (ໃຊ້ HTML, ລັນດ້ວຍ Web Browser) ຈຸດນີ້ອາດຈະຍັງບໍ່ເຫັນພາບ ດຽວຄ່ອຍໆເບິ່ງໄປພ້ອມກັນ.

#### ➤ ທີ່ມາຂອງ Progressive Web Apps

ມາດຕະຖານນີ້ຜູ້ເລີ່ມຕົ້ນຄື Google ນັ້ນເອງ ຈະເຫັນວ່າທຸກມື້ນີ້ຜູ້ໃຊ້ໂທລະສັບຕິດຕັ້ງແອບໃໝ່ ໜ້ອຍເຖິງຫຼາຍທີ່ສຸດລວມໄປເຖິງຖ້າເວັບຄອນເທັນເວັບໃດເວັບໜຶ່ງຕ້ອງການທີ່ຈະເຮັດ Application ເພື່ອໃຫ້ຜູ້ຕິດຕາມອ່ານ ກໍເບິ່ງຄ້າຍຄືຈະເປັນການລົງທຶນທີ່ຫຼາຍເກີນໄປ.

ຈຶ່ງມີແນວຄິດເຮັດແອບທີ່ຮັ້ນດ້ວຍ Web Browser ຊຶ່ງເບື້ອງໜຶ່ງຄືເວັບໄຊເດີມທີ່ເຮັດໄວ້ຢູ່ແລ້ວ ໂດຍນໍາ Logo ຂອງເວັບນັ້ນມາຢູ່ທີ່ໜ້າ Homescreen ເມື່ອກົດເຂົ້າໄປຈະພົບກັບເວັບໄຊທີ່ມີໜ້າຕາ ແລະ ຟີເຈີແບບແອບເລີຍ.

ຄິດພາບວ່າຈາກເດີມທີ່ເຮົາຈະຕ້ອງ ເຂົ້າ Store > ຄົ້ນຫາແອບ > ຖ້າດາວໂຫຼດ > ຕິດຕັ້ງ > ກົດຮັບ Permission > ເລີ່ມໃຊ້ງານ ຊຶ່ງຂັ້ນຕອນຈະຫຼຸດລົງເຫຼືອພຽງເຂົ້າເວັບ > ກົດ Add to Homescreen ແລ້ວໃຊ້ງານໄດ້ເລີຍ.

#### ➤ ຮູ້ຈັກກັບ Native App ແລະ Mobile Web App

Native App ຄື Mobile Application ທີ່ສ້າງມາຈາກພາສາທີ່ອອກແບບມາສໍາລັບ Platform ນັ້ນໆ ໂດຍສະເພາະ ສາມາດຂັບປະສົດທິພາບຂອງເຄື່ອງມືຕ່າງໆເທິງ Platform ນັ້ນມາໃຊ້ໄດ້ຢ່າງເຕັມທີ່ເຊັ່ນ: ພາສາ Swift ເທິງ iOS.

Mobile Web App ຄື Mobile Application ທີ່ສ້າງມາຈາກພາສາທີ່ໃຊ້ເຮັດເວັບໄຊ ຂໍ້ດີຄື ຂຽນເທື່ອດຽວສາມາດນໍາໄປສ້າງແອບເພື່ອຮັ້ນເທິງ iOS ແລະ Android ແຕ່ອາດຈະມີຂໍ້ຈຳກັດໃນບາງຟີເຈີ ຊຶ່ງປະຈຸບັນກໍພັດທະນາມາຂ້ອນຂ້າງຫຼາຍແລ້ວ.

ຍ້ອນກັບໄປເມື່ອປະມານ 6-7 ປີກ່ອນ ທີ່ນັກພັດທະນາເລີ່ມສົນໃຈການຂຽນໂປຣແກຣມເທິງໂທລະສັບ Platform ທີ່ນັກພັດທະນາຕ້ອງຈັບຕາເບິ່ງບໍ່ໄດ້ມີແຕ່ iOS ແລະ Android ແຕ່ຍັງມີພວກ Windows Phone, Black Berry ແລະ ອີກຫຼາຍເຈົ້າທີ່ເລັ່ງຈະມາຕິຕະຫຼາດ ເຮັດໃຫ້ເກີດບັນຫາ

Distribution Platforms ເຮັດໜຶ່ງແອບ ແຕ່ຕ້ອງລົງທຶນຂຽນຮອງຮັບທຸກເຈົ້າ ແລະ ນັກພັດທະນາໜຶ່ງຄົນຈະຕ້ອງຂຽນເປັນຫຼາຍພາສາ.

ຕັ້ງແຕ່ຕອນນັ້ນຈຶ່ງມີຄອນເຊບການ Cross Platform ເກີດຂຶ້ນ ໂດຍການສ້າງໂປຣແກຣມຂຶ້ນມາໜຶ່ງຕົວດ້ວຍພາສາໂປຣແກຣມທີ່ໃຊ້ເຮັດເວັບໄຊ ແລ້ວສາມາດນຳໂປຣແກຣມໄປໃຊ້ໄດ້ກັບທຸກ Platform ຊຶ່ງຕອນນັ້ນກໍມີເຈົ້າດັງໆ ເຊັ່ນ: PhoneGap ຫຼື Ionic ອອກມາ.

ແຕ່ດ້ວຍຄວາມທີ່ມັນສ້າງຈາກເທັກໂນໂລຢີເວັບ ແລະ ຍັງບໍ່ໄດ້ເປັນ Native ຂອງພາສາໂປຣແກຣມທີ່ອອກແບບມາສະເພາະທາງ ຈຶ່ງມີຂໍ້ຈຳກັດຫຼາຍຢ່າງທີ່ເຮັດໃຫ້ພັດທະນາໄດ້ບໍ່ສະດວກ ຊຶ່ງກໍພັດທະນາກັນມາເລື້ອຍໆຕາມການເວລາ ແລະ ເມື່ອ Platform ອື່ນມັນຕາຍໄປຈົນເຫຼືອແຕ່ສອງເຈົ້າກໍງ່າຍຂຶ້ນມາ.

ຈະບອກໄວ້ວ່າ PWA ນັ້ນ ບໍ່ແມ່ນທັງ Mobile Web, Native App ຫຼື Web Responsive (ເຖິງແມ່ນຈະຄ້າຍກັນພໍສົມຄວນ) ແຕ່ເອີ້ນວ່າເປັນມາດຕະຖານໃໝ່ຂອງການເຮັດເວັບທີ່ທັງໝົດຈຸດເດັ່ນຂອງທຸກຮູບແບບມາໃຊ້.

#### ➤ ພິເຈດີທຸກໂດຍທົ່ວໄປຂອງ Progressive Web Apps

- ໃຊ້ງານໄດ້ທຸກ Browser ແລະ ທຸກ Mobile Platform.
- ເຮັດ Cache ເກັບໄວ້ໃນໂຕ ນັກພັດທະນາຈະເປັນຄົນກຳນົດວ່າຈະ Cache ສິ່ງໃດໄວ້ແຕ່ເຮັດໃຫ້ສາມາດໃຊ້ງານໄດ້ທັງ Online ແລະ Offline.
- ມີການອັບເດດຂໍ້ມູນທັນທີເມື່ອ Online.
- Layout ຄືກັບ Application ຜູ້ໃຊ້ບໍ່ຕ້ອງຮຽນຮູ້ໃໝ່.
- ສາມາດ Push Notification ໄດ້ ໂດຍທີ່ບໍ່ຕ້ອງເປີດເວັບ ຫຼື ແອບຖິ້ມໄວ້.

ໃຜທີ່ສົນໃຈຕົວຢ່າງດ້ານເທິງສາມາດເຂົ້າໄປຫຼິ້ນໄດ້ທີ່ [www.pokedex.org](http://www.pokedex.org) ຫຼື ອີກເວັບໄຊໜຶ່ງທີ່ໃຫຍ່ [www.aliexpress.com](http://www.aliexpress.com) ລອງກົດ Add to Home Screen ແລ້ວປິດອິນເຕີເນັດເລີຍ (ຕອນນີ້ສະເພາະເທິງ Android ເທົ່ານັ້ນ)

ລວມເວັບໄຊອື່ນໆ ທີ່ເປັນ PWA : [pwa.rocks](http://pwa.rocks)

ອະນາຄົດຂອງ Progressive Web Apps

ໃນສ່ວນອະນາຄົດຂອງ PWA ນັ້ນ Google Developers Expert ດ້ານ Web Technologies ທີ່ໄກ້ຊິດກັບເທັກໂນໂລຢີໃນຝັ່ງເວັບຂອງ Google ໄດ້ໃຫ້ຄວາມເຫັນເອົາໄວ້ວ່າ

PWA ເປັນສິ່ງທີ່ສາມາດຮັບໃຊ້ experience ໃຫ້ຜູ້ໃຊ້ໄດ້ຄືກັບແອັບແຕ່ເຮັດວຽກຢູ່ເທິງ Browser ຊຶ່ງຈະຫຼຸດຊັ້ນຕອນຄວາມຫຍຸ້ງຍາກເທິງ Store ແລະ ໃນສ່ວນຂອງນັກພັດທະນານັ້ນສາມາດເຮັດໄດ້ໃນຕົ້ນທຶນທີ່ຖືກກວ່າເຮັດແອັບຫຼາຍເທົ່າ.

ໃນອະນາຄົດນັ້ນ PWA ຈະຖືກນຳມາໃຊ້ເປັນແອັບນ້ອຍໆ ບໍ່ໄດ້ມີ Interactive ຫຼາຍລວມໄປເຖິງເຮັດເປັນ Lite Version ແຍກອອກມາຈາກແອັບໃຫຍ່ ເຊັ່ນ: uber ທີ່ມີແອັບຫຼັກຢູ່ ແຕ່ກໍມີ PWA ແຍກອອກມາທີ່ຜູ້ໃຊ້ສາມາດເຂົ້າເວັບແລ້ວກົດຈອງໄດ້ຢ່າງວ່ອງໄວບໍ່ຈຳເປັນຕ້ອງຜ່ານຂະບວນການຫຍຸ້ງຍາກໃນການໂຫຼດແອັບເທິງ Store ຫຼື aliexpress.com ທີ່ສາມາດເບິ່ງສິນຄ້າ ແລະ ເລືອກຊື້ໄດ້ຢ່າງວ່ອງໄວ.

ໃນສ່ວນຂອງ iOS ນັ້ນຂ້ອນຂ້າງເປັນເລື່ອງໃຫຍ່ເພາະຖ້າຝັ່ງນັ້ນບໍ່ເອົານຳກໍແຈ້ງເກີດຍາກຢ່າງໃດກໍຕາມຕໍ່ໃຫ້ບໍ່ໄດ້ Support ເວັບທີ່ເປັນ PWA ກໍສາມາດໃຊ້ງານເທິງເວັບໄຊໄດ້ປົກກະຕິບໍ່ຄາບັນຫາຫຍັງ.

ຕອນນີ້ລາຍໃຫຍ່ຫຼາຍເຈົ້າໃຫ້ຄວາມສຳຄັນກັບ PWA ກັນຫຼາຍ ແລະ ເລີ່ມດຳເນີນການໄປແລ້ວ ເຊັ່ນ: Tinder, Lyft, aliexpress, uber ສຳລັບ Social Network ລາຍໃຫຍ່ຢ່າງ Facebook ແລະ Twitter ກຳລັງຢູ່ໃນຊ່ວງທົດສອບ ສ່ວນ Instagram ເຮັດອອກມາຮຽບຮ້ອຍ ແລະ ຟີເຈີ.

### 2.1.13 ຄວາມຮູ້ກ່ຽວກັບໂປຣແກຣມ Microsoft Office 2016

Microsoft word ແມ່ນໂປຣແກຣມປະມວນຜົນເຊິ່ງອອກແບບມາເພື່ອຊ່ວຍໃຫ້ສ້າງເອກະສານທີ່ມີຄຸນນະພາບໃນລະດັບມືອາຊີບ ເຄື່ອງມືການຈັດຮູບແບບເອກະສານທີ່ດີທີ່ສຸດຂອງ word ຈະເຮັດໃຫ້ສາມາດຈັດລະບຽບ ແລະ ຂຽນເອກະສານຂອງທ່ານໄດ້ຢ່າງມີປະສິດທິພາບຫຼາຍຂຶ້ນ word ຍັງມີເຄື່ອງມືການແກ້ໄຂ ແລະ ກວດສອບຄຳຜິດທີ່ມີປະສິດທິພາບ ສາມາດເຮັດວຽກຮ່ວມກັບຜູ້ອື່ນໄດ້ຢ່າງງ່າຍດາຍ.

#### 1) ປະໂຫຍດຂອງໂປຣແກຣມ Microsoft Word Office 2016

- ມີລະບົບປະຕິບັດຕ່າງໆທີ່ຊ່ວຍໃນການເຮັດວຽກໃຫ້ສະດວກສະບາຍຂຶ້ນເຊັ່ນ: ການກວດຄຳສະກົດ ການກວດສອບໄວຍະກອນ, ການໃສ່ຂໍ້ຄວາມອັດຕະໂນມັດເປັນຕົ້ນ.
- ສາມາດໃຊ້ສ້າງຈົດໝາຍໄດ້ຢ່າງວ່ອງໄວ ໂດຍສາມາດກຳນົດໃຫ້ຜູ້ວິເສດ( Wizard )ໃນ word ສ້າງແບບຟອມຂອງຈົດໝາຍໄດ້ຫຼາຍຮູບແບບຕາມທີ່ຕ້ອງການ.
- ສາມາດໃຊ້ word ສ້າງຕາຕະລາງທີ່ສະຫຼັບຊັບຊ້ອນໄດ້.
- ປັບປຸງເອກະສານໄດ້ງ່າຍ ແລະ ວ່ອງໄວ ສາມາດຕົກແຕ່ງເອກະສານ ຫຼື ເພື່ອຄວາມສະດວກ.
- ຈະໃຫ້ word ສາມາດແຊກຮູບພາບ, ກຣາບຟິກ ຫຼື ຜັງອົງກອນລົງໃນເອກະສານໄດ້.
- word ປັບແຕ່ງໃຫ້ກໍໄດ້ ໂດຍສາມາດເປັນຜູ້ກຳນົດຮູບແບບຂອງເອກະສານເອງ.

- ຄວາມສາມາດໃນການເຊື່ອມຕໍ່ໂປຣແກຣມອື່ນໆ ໃນຊຸດໂປຣແກຣມ Microsoft office ສາມາດໂອນຍ້າຍຂໍ້ມູນຕ່າງໆລະຫວ່າງໂປຣແກຣມໄດ້.
- ສ້າງເອກະສານໃຫ້ໃຊ້ວຽກໃນອິນເຕີເນັດໄດ້ຢ່າງງ່າຍດາຍ.

## 2) ຄວາມສາມາດຂອງ Microsoft word

- ສາມາດພິມ ແລະ ແກ້ໄຂເອກະສານ.
- ສາມາດລຶບ, ຍ້າຍ ແລະ ຮ່າງຂໍ້ຄວາມ.
- ສາມາດພິມຕົວອັກສອນປະເພດຕ່າງໆ.
- ສາມາດຂະຫຍາຍຕົວອັກສອນ.
- ສາມາດຈັດຕົວອັກສອນໃຫ້ເປັນຕົວໜ້າ,ຕົວຫຼັງ ແລະ ຂີດກ້ອງໄດ້.
- ສາມາດໃສ່ເຄື່ອງໝາຍ ແລະ ຕົວເລກລຳດັບຕາມຫົວຂໍ້.
- ສາມາດແປງເສົາໄດ້.
- ສາມາດກວດການສະກົດ ແລະ ແກ້ໄຂໃຫ້ຖືກຕ້ອງ.
- ສາມາດຄົ້ນຫາ ແລະ ປຸງແປງຂໍ້ຄວາມທີ່ພິມຜິດ.
- ສາມາດຈັດຂໍ້ຄວາມເອກະສານໃຫ້ພິມໄປທາງຊ້າຍ, ທາງຂວາ ແລະ ລະຫວ່າງກາງ.
- ສາມາດໃສ່ຮູບພາບໃນເອກະສານ.
- ສາມາດຕົກແຕ່ງໂຕອັກສອນ,ພິມຕາຕະລາງ.

### 2.1.14 ຄວາມຮູ້ກ່ຽວກັບໂປຣແກຣມ Microsoft Visio 2016

ໂປຣແກຣມ Microsoft Visio ເປັນເຄື່ອງມືເສີມການເຮັດວຽກຂອງ Microsoft Office ໃນການສ້າງແຜນວາດ (Diagram) ປະເພດຕ່າງໆ, ເຊິ່ງເຮັດໄດ້ງ່າຍດາຍ, ສະດວກ ແລະ ວ່ອງໄວ. ເປັນທີ່ນິຍົມໃຊ້, ມີປະໂຫຍດຫຼາຍສຳລັບການຈັດເອກະສານ, ລວມໄປເຖິງແຜນວາດການອອກແບບ, ຂັ້ນຕອນເຮັດການວິເຄາະ ແລະ ອອກກແບບລະບົບຕ່າງໆ.

ສຳລັບໂປຣແກຣມ Microsoft Visio ມີໃຫ້ເລືອກຢູ່ 2 ປະເພດຄື:

- 1) Microsoft Visio Standard ເໝາະກັບວຽກດ້ານທຸລະກິດເຊັ່ນ: ຜູ້ບໍລິຫານໂຄງການ, ນັກການຕະຫຼາດ, ພະນັກງານຝ່າຍຊັບພະຍາກອນມະນຸດ ແລະ ທີມງານມີໜ້າທີ່ເບິ່ງແຍງການດຳເນີນງານເພື່ອຊ່ວຍໃນການເບິ່ງແຜນວາດ ແລະ ຂ່າວສານ.
- 2) Microsoft Visio Professional ເໝາະສຳລັບມືອາຊີບທາງດ້ານເຕັກນິກ, ພະນັກງານໄອທີ, ນັກພັດທະນາ ແລະ ວິສະວະກອນທີ່ຊ່ວຍໃນການອອກແບບຂໍ້ມູນລະບົບ ເພື່ອໃຊ້ໃນການເຮັດຕົ້ນແບບ Microsoft Visio Professional ເປັນໂປຣແກຣມ ທີ່ຖືກສ້າງຂຶ້ນມາເພື່ອຊ່ວຍໃນການສ້າງ Flow chart ຫຼື Diagram ຂອງວຽກໃນສາຂາຕ່າງໆໃຫ້ເຮັດວຽກໄດ້ງ່າຍຂຶ້ນ. ລັກສະນະຢ່າງ

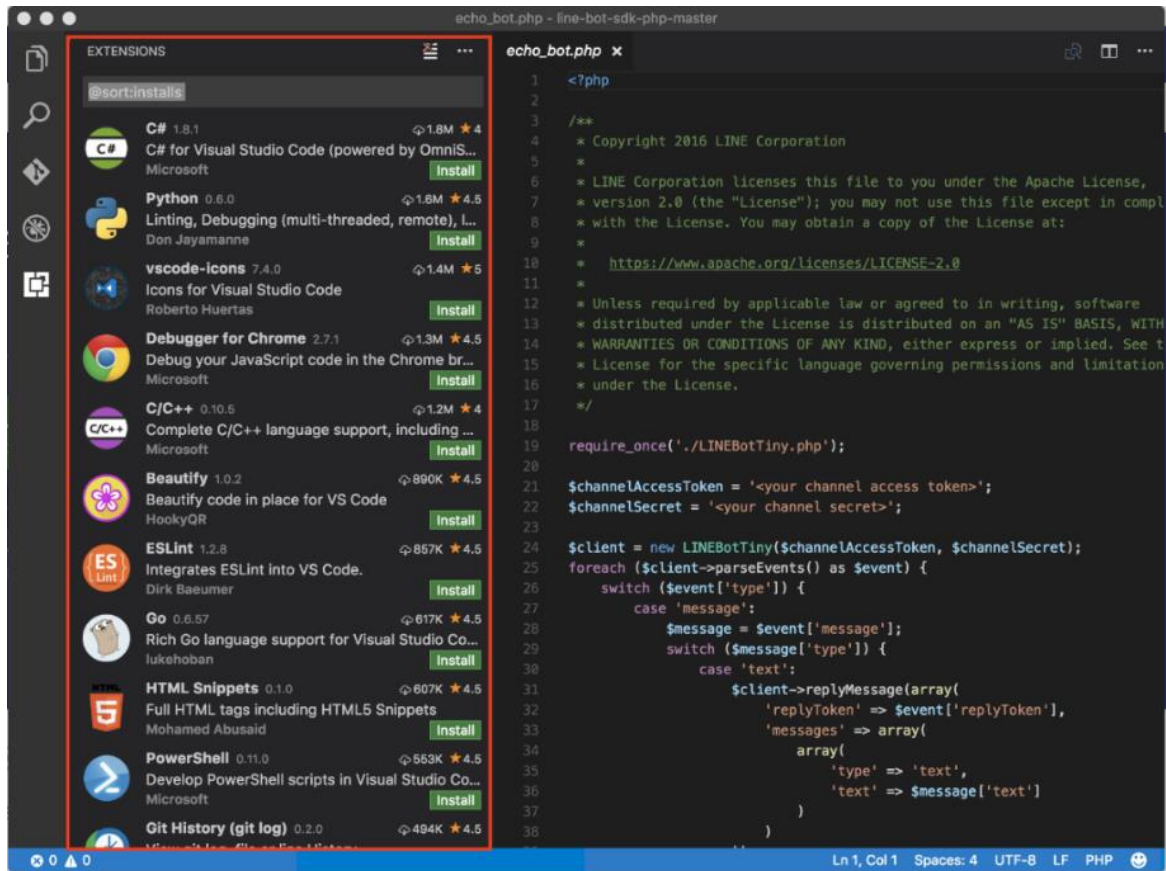
ໜຶ່ງໃນການສ້າງ Flow chart ໃນ Microsoft Visio Professional ກໍຄືມີຮູບແບບ Diagram ພື້ນຖານຕ່າງໆ ຈັດກຸ່ມໄວ້ໃຫ້, ເຊິ່ງງ່າຍໃນການອອກແບບ ແລະ ໃຊ້ງານ.

ຈຸດດີ Microsoft Visio ຄື: ຊອກຫາຂໍ້ຜິດພາດໄດ້ຢ່າງສະດວກ ແລະ ເຂົ້າໃຈງ່າຍໃນການສະແດງທິດທາງການໄຫຼຂໍ້ມູນ.

### 2.1.15 ຄວາມຮູ້ກ່ຽວກັບໂປຣແກຣມ Visual Studio Code

Visual Studio Code ຫຼື ທີ່ຫຼາຍຄົນນິຍົມເອີ້ນຫຍໍ້ວ່າ “vs code” ບອກກ່ອນວ່າ Editor ໂຕນີ້ມັນອອກມາຕັ້ງແຕ່ 29 ເມສາ ປີ 2015 ແລ້ວພັດທະນາຂຶ້ນໂດຍບໍລິສັດຍັກໃຫຍ່ໄມໂຄຣຊອບ(Microsoft) ເປັນທັງໂຕແກ້ໄຂ ແລະ ປັບແຕ່ງໂຄດ (code optimized editor) ທີ່ຕັດຄວາມສາມາດມາຈາກ Visual Studio ລຸ້ນປົກກະຕິ (ພວກ GUI designer) ອອກໄປເຫຼືອແຕ່ໂຕ editor ຢ່າງດຽວ ສາມາດເຮັດວຽກໄດ້ຂ້າມແພລດຟອມທັງໝົດວິນ Windows, Mac ແລະ Linux ຊັບພອດພາສາຫຼາຍຮ້ອຍພາສາອີກ ຊຶ່ງທາງໄມໂຄຊອບເອງນັ້ນໄດ້ເປີດໃຫ້ໃຊ້ຟຣີອີກດ້ວຍ.

ຄວາມສາມາດຂອງ “vs code” ນັ້ນຈະມີຄວາມສາມາດໃນການເປີດໄດ້ຄືກັບ editor ໂຕອື່ນໆ ເຊັ່ນ: sublime, Atom, Notepad++ທັງໝົດເຖິງຄວາມສາມາດໃນການຕິດຕັ້ງເຄື່ອງມືເສີມ (Extension) ໂດຍຮັບຮອງໄດ້ວ່າມີຊັບພອດຢ່າງແນ່ນອນ ເພາະວ່າມັນຖືກພັດທະນາມາໃຫ້ຕອບໂຈດນັກພັດທະນາຫຼາຍທີ່ສຸດ ແມ່ນການດຶງຊ້າຍໜ້າຕາ ໃຫ້ເປັນຮູບແບບທີ່ເຂົ້າໃຈ ແລະ ໃຊ້ງານໄດ້ງ່າຍ ບໍ່ຕ້ອງສຶກສາຫຍັງເພີ່ມເຕີມກໍໃຊ້ງານໄດ້ເລີຍ ສິ່ງທີ່ເຮັດໃຫ້ມັນໂດດເດັ່ນກວ່າໂຕອື່ນໆ ຄືການທີ່ອອກແບບໃຫ້ການຄົ້ນຫາສິ່ງຕ່າງໆ ເຮັດອອກມາໃຫ້ໃຊ້ງານໄດ້ງ່າຍ ແລະ ເບິ່ງງ່າຍກວ່າໂຕອື່ນໆ ການທີ່ສ້າງໃຫ້ສາມາດເຊື່ອມຕໍ່ກັບ Git ໄດ້ຢ່າງວ່ອງໄວ ແລະ ງ່າຍດາຍ ມີຟັງຊັນໃນການ commit, push & pull ຢູ່ໃນໂຕ ຫຼື ຈະເບິ່ງ change ຂອງຟາຍທີ່ເກີດຂຶ້ນກໍໄດ້ແບບງ່າຍດາຍບໍ່ຕ້ອງສຶກສາຫຍັງເພີ່ມແຕ່ກໍໃຊ້ງານໄດ້ເລີຍ.



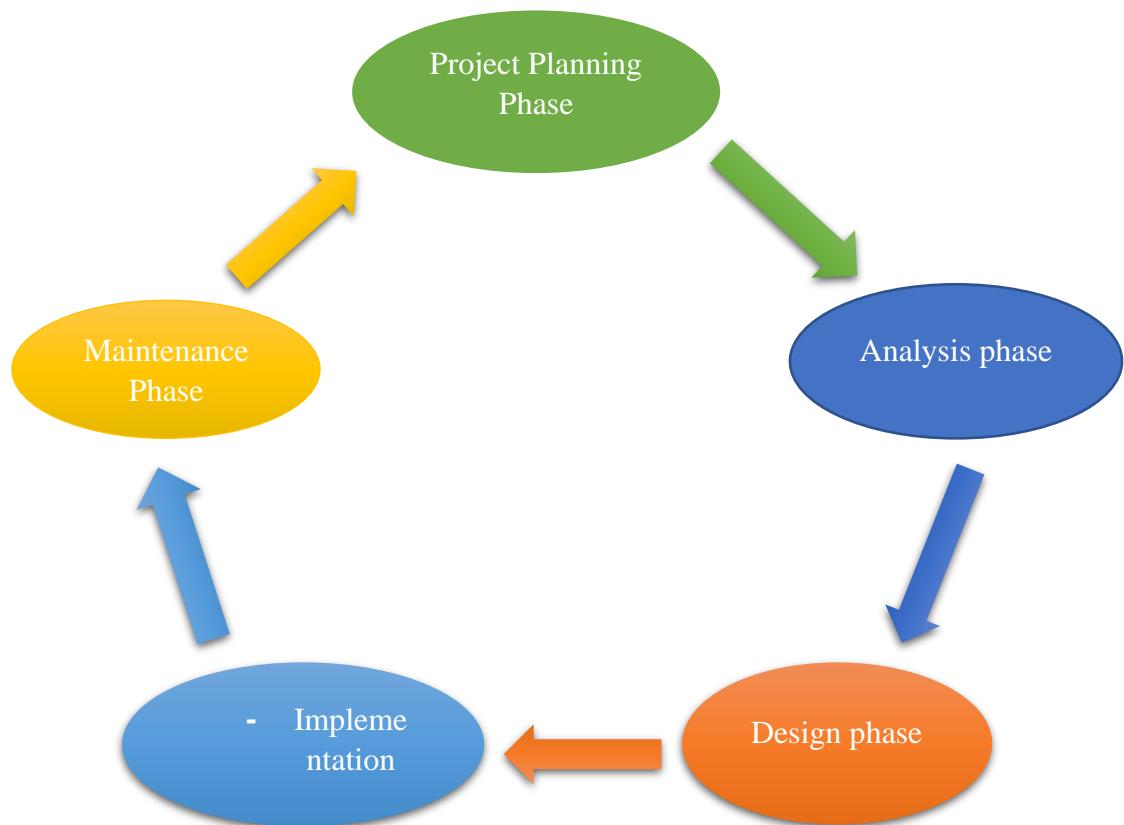
រូបភាព ៩ ប្រព័ន្ធ VSCode

### ບົດທີ 3

#### ວິທີດໍາເນີນການຄົ້ນຄວ້າ

##### 3.1 ວິທີສຶກສາ ແລະ ຄົ້ນຄວ້າ

ເປັນຂະບວນການທີ່ສະແດງເຖິງການດໍາເນີນຂັ້ນຕອນການເຮັດວຽກຂອງລະບົບຕັ້ງແຕ່ຕົ້ນຈົນຈົບ, ບົດໂຄງການຈົບຊັ້ນນີ້ພວກຂ້າພະເຈົ້າໄດ້ນຳໃຊ້ທິດສະດີການວິເຄາະ ແລະ ອອກແບບລະບົບແບບໂຄງທີ່ປະກອບມີ 5 ໄລຍະຄື:



ແຜນວາດທີ 1 : ແຜນວາດລວມຂອງລະບົບ



- ໄລຍະທີ່ 1 ການວາງແຜນໂຄງການ (Project Planning phase)
- ໄລຍະທີ່ 2 ການວິເຄາະ (Analysis phase)
- ໄລຍະທີ່ 3 ການອອກແບບ (Design phase)
- ໄລຍະທີ່ 4 ການນຳໄປໄຊ້ (Implementation phase)
- ໄລຍະທີ່ 5 ການບຳລຸງຮັກສາ (Maintenance phase)

### 3.1.1 ໄລຍະທີ່ 1 ການວາງແຜນໂຄງການ (Project planning phase)

ການວາງແຜນໂຄງການຈັດເປັນຂະບວນການພື້ນຖານໃນຄວາມເຂົ້າໃຈຢ່າງເລິກເຊິ່ງວ່າເປັນ ທັງຕ້ອງສ້າງລະບົບໃໝ່ ທີມງານຕ້ອງພິຈາລະນາວ່າຈະຕ້ອງດຳເນີນງານຕໍ່ໄປແນວໃດກ່ຽວກັບ ຂະບວນການສ້າງລະບົບໃໝ່ ກ່ອນອື່ນໝົດຕ້ອງມີຈຸດກຳເນີດຂອງລະບົບງານເຊິ່ງໂດຍປົກກະຕິແລ້ວ ຈຸດກຳເນີດຂອງລະບົບງານມັກເກີດຂຶ້ນຈາກຜູ້ໃຊ້ລະບົບເປັນຜູ້ທີ່ຕິດແທດກັບລະບົບໂດຍກົງເຮັດໃຫ້ມີ ຄວາມໃກ້ສືດກັບລະບົບວຽກທີ່ດຳເນີນຢູ່ຫຼາຍທີ່ສຸດ ເມື່ອຜູ້ໃຊ້ລະບົບມີຄວາມຕ້ອງການປັບປຸງລະບົບ ງານ ດັ່ງນັ້ນຈຶ່ງຖືເປັນຈຸດເລີ່ມຕົ້ນໃນບົດບາດຂອງນັກວິເຄາະລະບົບວ່າຈະຕ້ອງສຶກສາເຖິງຂອບເຂດ ຂອງບັນຫາທີ່ຜູ້ໃຊ້ລະບົບກຳລັງປະສົບບັນຫາຢູ່ ແລະ ຈະດຳເນີນການແກ້ໄຂແນວໃດ ສຶກສາເຖິງ ຄວາມເປັນໄປໄດ້ວ່າລະບົບໃໝ່ທີ່ຈະພັດທະນາຂຶ້ນມານັ້ນມີຄວາມເປັນໄປໄດ້ ແລະ ຄຸ້ມຄ່າກັບການຈະ ລົງທຶນ ຫຼື ບໍ່.

ແນວໃດກໍ່ຕາມໄລຍະຂອງການວາງແຜນໂຄງການປົກກະຕິມັກຈະມີໄລຍະເວລາທີ່ສັ້ນ ແຕ່ກໍ່ຖື ວ່າເປັນຂັ້ນຕອນທີ່ສຳຄັນທີ່ຈະໃຫ້ເກີດຜົນສຳເລັດໄດ້ ດັ່ງນັ້ນໃນໄລຍະຂອງການວາງແຜນໂຄງການ ຈຶ່ງໄດ້ອາໄສນັກວິເຄາະລະບົບທີ່ມີຄວາມຮູ້ ແລະ ປະສົບການສູງ ເນື່ອງຈາກວ່າຫາກນັກວິເຄາະ ລະບົບບໍ່ເຂົ້າໃຈເຖິງບັນຫາອັນແທ້ຈິງທີ່ເກີດຂຶ້ນ ກໍ່ຈະບໍ່ສາມາດພັດທະນາລະບົບຂຶ້ນມາເພື່ອແກ້ໄຂ ບັນຫາໃຫ້ຖືກຈຸດໄດ້ ແລະ ມັກຈະມີໂຄງການພັດທະນາລະບົບຫຼາຍໂຄງການທີ່ຫຼັງຈາກໄດ້ດຳເນີນ ການພັດທະນາ ແລະ ນຳມາໃຊ້ງານແລ້ວປະກົດວ່າບໍ່ສາມາດຕອບສະໜອງຄວາມຕ້ອງການຂອງຜູ້ ໃຊ້ງານເຊິ່ງຖືວ່າເປັນເລື່ອງທີ່ກໍ່ໃຫ້ເກີດຄວາມສູນເສຍທັງທາງດ້ານການລົງທຶນ ແລະ ໄລຍະເວລາ.

ສະຫຼຸບລວມແລ້ວໄລຍະຂອງການວາງແຜນໂຄງການປະກອບມີກິດຈະກຳຕ່າງໆດັ່ງນີ້:

- ກຳນົດບັນຫາ Problem Definition.
- ສຶກສາຄວາມເປັນໄປໄດ້ຂອງໂຄງການ Feasibility study.
- ສ້າງຕາຕະລາງກຳນົດເວລາໂຄງການ Project scheduling

- ຈັດຕັ້ງທີມງານໂຄງການ Staff the project

### 3.1.2 ໄລຍະທີ 2 ການວິເຄາະ (Analysis Phase)

ໄລຍະການວິເຄາະຈະຕ້ອງມີຄໍາຕອບກ່ຽວກັບຄໍາຖາມວ່າໃຜເປັນຜູ້ທີ່ໃຊ້ລະບົບ ແລະ ມີຫຍັງແດ່ທີ່ຈະຕ້ອງເຮັດໃນໄລຍະນີ້ນັກວິເຄາະລະບົບຈະຕ້ອງດໍາເນີນການໃນຂັ້ນຕອນຂອງການວິເຄາະລະບົບງານປັດຈຸບັນ (Current system) ເພື່ອນໍາມາພັດທະນາແນວຄວາມຄິດສໍາລັບລະບົບໃໝ່ (New System).

ຈຸດປະສົງຫຼັກຂອງການວິເຄາະຄືຈະຕ້ອງສຶກສາ ແລະ ສ້າງຄວາມເຂົ້າໃຈໃນຄວາມຕ້ອງການຕ່າງໆທີ່ໄດ້ລວບລວມມາ ດັ່ງນັ້ນ ການລວບລວມຄວາມຕ້ອງການ (Requirements Gathering) ຈຶ່ງຈັດເປັນວຽກພື້ນຖານຂອງການວິເຄາະລະບົບໂດຍຂໍ້ມູນຄວາມຕ້ອງການເຫຼົ່ານີ້ນັກວິເຄາະລະບົບຈະນໍາມາວິເຄາະເພື່ອທີ່ຈະປະເມີນວ່າຄວນມີຫຍັງແດ່ທີ່ລະບົບໃໝ່ຕ້ອງດໍາເນີນການ ແລະ ດ້ວຍເຫດນີ້ເອງການກໍານົດລາຍລະອຽດກ່ຽວກັບຄວາມຕ້ອງການຂອງຜູ້ໃຊ້ (User Requirements) ຈະເພີ່ມຄວາມສໍາຄັນຫຼາຍຂຶ້ນເປັນລໍາດັບສໍາລັບລະບົບທີ່ມີຄວາມສັບຊ້ອນສູງ ແລະ ນັກວິເຄາະຕ້ອງເອົາໃນໃສ່ກັບການລວບລວມຄວາມຕ້ອງການຈາກຜູ້ໃຊ້ບໍ່ຄວນກໍານົດຄວາມຕ້ອງການຂຶ້ນເອງໂດຍໃຊ້ຄວາມຄິດສ່ວນຕົວຂອງຕົນເອງເປັນຫຼັກ ຫຼື ປະເມີນຄວາມຕ້ອງການຂອງຜູ້ໃຊ້ລະບົບບໍ່ເຂົ້າກັບຈຸດປະສົງ ແລະ ຫາກມີການພັດທະນາລະບົບຕໍ່ໄປຈົນແລ້ວລະບົບວຽກທີ່ໄດ້ກໍ່ຈະບໍ່ກົງກັບຄວາມຕ້ອງການຂອງຜູ້ໃຊ້ລະບົບຢ່າງແທ້ຈິງເຮັດໃຫ້ຕ້ອງມີຄວາມປັບປຸງ ຫຼື ປ່ຽນແປງພາຍຫຼັງ.

ນັກວິເຄາະລະບົບສາມາດລວບລວມຄວາມຕ້ອງການຕ່າງໆໄດ້ຈາກການສັງເກດ, ການເຮັດວຽກຂອງຜູ້ໃຊ້, ການໃຊ້ເຕັກນິກ, ການສຳພາດ ຫຼື ການໃຊ້ແບບສອບຖາມການອ່ານເອກະສານກ່ຽວກັບການປະຕິບັດງານຂອງລະບົບວຽກ ປັດຈຸບັນລະບຽບກົດເກນຂອງບໍລິສັດ ແລະ ການມອບໝາຍຕໍາແໜ່ງໜ້າທີ່ຮັບຜິດຊອບເຊິ່ງໃນຊ່ວງຂອງການເກັບກໍາຂໍ້ມູນຄວາມຕ້ອງການກໍ່ຈະພົບກັບຜູ້ໃຊ້ໃນລະດັບຕ່າງໆທີ່ເຮັດໃຫ້ຮູ້ເຖິງບັນຫາ ແລະ ແນວທາງການແກ້ໄຂບັນຫາທີ່ແນະນໍາໂດຍຜູ້ໃຊ້ ດັ່ງນັ້ນ ການເກັບກໍາຄວາມຕ້ອງການຈຶ່ງເປັນກິດຈະກຳທີ່ສໍາຄັນເພື່ອຄົ້ນຫາຄວາມຈິງ ແລະ ຕ້ອງສ້າງຄວາມຕ້ອງການເຊິ່ງກັນ ແລະ ກັນ ເພື່ອສະຫຼຸບອອກມາເປັນຂໍ້ກໍານົດທີ່ມີຄວາມຊັດເຈນໂດຍຂໍ້ກໍານົດເຫຼົ່ານີ້ເມື່ອຜູ້ໃຊ້ທີ່ກ່ຽວຂ້ອງໄດ້ອ່ານແລ້ວຈະຕ້ອງສຶກສາຄວາມໝາຍໄດ້ກົງກັນ.

ຫຼັງຈາກໄດ້ນໍາຄວາມຕ້ອງການຕ່າງໆມາສະຫຼຸບເປັນຂໍ້ກໍານົດທີ່ຊັດເຈນແລ້ວຂັ້ນຕອນຕໍ່ໄປກໍ່ຄືນໍາຂໍ້ກໍານົດເຫຼົ່ານັ້ນໄປພັດທະນາອອກມາເປັນຄວາມຕ້ອງການຂອງລະບົບໃໝ່ ໂດຍເຕັກນິກທີ່ໃຊ້ກໍ່ຄື

ການພັດທະນາແບບຈຳລອງຂະບວນການ (Process Model) ເຊິ່ງເປັນແຕ່ ນພາບທີ່ໃຊ້ອະທິບາຍເຖິງ ຂະບວນການທີ່ຕ້ອງດຳເນີນໃນລະບົບວ່າມີຫຍັງແດ່ ແລະ ຕໍ່ໄປກໍ່ດຳເນີນການພັດທະນາແບບຈຳລອງ ຂໍ້ມູນ(Data Model) ເພື່ອອະທິບາຍເຖິງຂໍ້ມູນທີ່ຈັດເກັບໄວ້ສຳລັບສະໜັບສະໜູນການເຮັກວຽກຕ່າງໆ.

ສະຫຼຸບໄລຍະຂອງການວິເຄາະລະບົບປະກອບມີກິດຈະກຳຕ່າງໆດັ່ງນີ້:

- ວິເຄາະລະບົບງານປັດຈຸບັນ.
- ເກັບກຳຄວາມຕ້ອງການໃນດ້ານຕ່າງໆ ແລະ ນຳມາວິເຄາະເພື່ອສະຫຼຸບເປັນຂໍ້ກຳນົດຊັດເຈນ.
- ນຳຂໍ້ກຳນົດມາພັດທະນາອອກມາເປັນຄວາມຕ້ອງການຂອງລະບົບໃໝ່.
- ສ້າງແບບຈຳລອງຂະບວນການຂອງລະບົບໃໝ່ໂດຍການແຕ້ມແຜນພາບກະແສຂໍ້ມູນ (Data Flow Diagram: DFD).
- ສ້າງແບບຈຳລອງຂໍ້ມູນໂດຍການແຕ້ມ Entity Relationship Diagram: ERD.

### 3.1.3 ໄລຍະທີ 3 ການອອກແບບ (Design Phase)

ໄລຍະການອອກແບບເປັນການພິຈາລະນາວ່າລະບົບລະດຳເນີນການໄປໄດ້ແນວໃດ ເຊິ່ງກ່ຽວ ຂ້ອງກັບຍຸດທະວິທີການຂອງການອອກແບບທີ່ວ່າດ້ວຍການຕັດສິນໃຈວ່າຈະພັດທະນາລະບົບໃໝ່ດ້ວຍ ແນວທາງໃດເຊັ່ນ ພັດທະນາຂຶ້ນເອງ, ຊື້ໂປຣແກຣມສຳເລັດຮູບ ຫຼື ວ່າຈ້າງບໍລິສັດພັດທະນາລະບົບໃຫ້ ເປັນຕົ້ນ. ນອກຈາກນີ້ໄລຍະການອອກແບບຈະກ່ຽວຂ້ອງກັບການອອກແບບທາງດ້ານສະຖາປັດຕະຍະ ກຳລະບົບທີ່ກ່ຽວຂ້ອງກັບອຸປະກອນຮາດແວ, ຊອບແວ ແລະ ເຄືອຂ່າຍ.

ການອອກແບບລາຍງານ (Out Design) ການອອກແບບໜ້າຈໍເພື່ອປະຕິສຳພັນກັບຜູ້ໃຊ້ (User Interface), ການອອກແບບຜັງງານລະບົບ (System Flowchart), ເຊິ່ງລວມເຖິງລາຍ ລະອຽດຂອງໂປຣແກຣມ (Specific Program), ຖານຂໍ້ມູນ (Database) ແລະ ຂໍ້ມູນທີ່ກ່ຽວຂ້ອງແນວໃດ ກໍ່ຕາມເຖິງວ່າກິດຈະກຳບາງສ່ວນຂອງໄລຍະອອກແບບນີ້ ສ່ວນໃຫຍ່ຈະຖືກດຳເນີນການໄປບາງສ່ວນ ແລ້ວໃນໄລຍະຂອງການວິເຄາະແຕ່ໄລຍະການອອກແບບນີ້ຈະເນັ້ນເຖິງການດຳເນີນການແກ້ໄຂ ບັນຫາແນວໃດຫຼາຍກ່ວາໂດຍການນຳຜົນຂອງແບບຈຳລອງທາງ Logical Model. ທີ່ໄດ້ຈາກການ ວິເຄາະມາພັດທະນາມາເປັນແບບຈຳລອງທາງ Physical Model.

- ການວິເຄາະຈະເນັ້ນແກ້ໄຂບັນຫາຫຍັງແດ່.
- ການອອກແບບຈະເນັ້ນການແກ້ໄຂບັນຫາແນວໃດ.

- ສະຫຼຸບໄລຍະການອອກແບບ.
- ພິຈາລະນາແນວທາງໃນການພັດທະນາລະບົບ.
- ອອກແບບສະຖາປັດຕະຍະກຳລະບົບ (Architecture Design).
- ອອກແບບຖານຂໍ້ມູນ (Database Design).
- ອອກແບບການສະແດງຜົນ (Output Design).
- ອອກແບບການປ້ອນຂໍ້ມູນ (Input Design).
- ອອກແບບສ່ວນຕິດຕໍ່ກັບຜູ້ໃຊ້ (User Interface).
- ສ້າງຕົ້ນແບບ (Prototype).
- ອອກແບບໂປຣແກຣມ (Structure Chart).

### 3.1.4 ໄລຍະທີ 4 ການນຳໄປໃຊ້ (Implementation Phase)

ໃນໄລຍະການນຳໄປໃຊ້ຈະເຮັດໃຫ້ລະບົບເກີດຜົນຂຶ້ນມາໂດຍການສ້າງລະບົບທົດສອບລະບົບ ແລະ ການຕິດຕັ້ງລະບົບໂດຍຈຸດປະສົງຫຼັກຂອງກິດຈະກຳໃນໄລຍະນີ້ ບໍ່ແມ່ນພຽງຄວາມໜ້າເຊື່ອຖືຂອງລະບົບ ຫຼື ລະບົບສາມາດເຮັດວຽກໄດ້ດີເທົ່ານັ້ນ ແຕ່ຕ້ອງໝັ້ນໃຈວ່າຜູ້ໃຊ້ລະບົບຕ້ອງໄດ້ຮັບການເຝິກອົບຮົມເພື່ອໃຊ້ງານລະບົບ ແລະ ຄວາມຄາດຫວັງໃນອົງກອນທີ່ຕ້ອງການຜົນຕອບແທນໃນດ້ານດີກັບການໃຊ້ລະບົບໃໝ່ລຳດັບກິດຈະກຳຕ່າງໆ ທຸກກິດຈະກຳຕ້ອງເຂົ້າມາດຳເນີນການຮ່ວມກັນໃນໄລຍະນີ້ເພື່ອໃຫ້ລະບົບການປະຕິບັດງານໄດ້ຮັບຄວາມປະສົບຜົນສຳເລັດໄດ້ໂດຍດີ.

ສະຫຼຸບໄລຍະການນຳໄປໃຊ້ຈະປະກອບມີກິດຈະກຳຕ່າງໆດັ່ງຕໍ່ໄປນີ້:

- ສ້າງລະບົບຂຶ້ນມາດ້ວຍການຂຽນໂປຣແກຣມ.
- ກວດສອບຄວາມຖືກຕ້ອງທາງດ້ານ Verification ແລະ Validation ແລະ ດຳເນີນການທົດສອບລະບົບ.
- ແປງຂໍ້ມູນ (Convert Data).
- ຕິດຕັ້ງລະບົບ (System Installation) ແລະ ສ້າງເອກກະສານຄູ່ມື.
- ເຝິກອົບຮົມຜູ້ໃຊ້ ແລະ ປະເມີນຜົນລະບົບໃໝ່.

ສຳລັບການສ້າງລະບົບ ຫຼື ການຂຽນໂປຣແກຣມນັ້ນ ສາມາດໃຊ້ວິທີການຂຽນໂປຣແກຣມດ້ວຍພາສາຄອມພິວເຕີເຊັ່ນ: ການໃຊ້ພາສາ Visual Basic, C#, PHP, Java... ນອກຈາກນີ້ ຍັງມີເຕັກນິກອື່ນໆເຊັ່ນ: ເຄື່ອງມືໃນການພັດທະນາ Application ເຊິ່ງເປັນຊອບແວຮີທີ່ເປັນແຫຼ່ງລວມຂອງ

ເຄື່ອງມືຕ່າງໆທີ່ໃຊ້ເພື່ອພັດທະນາ Application ເຮັດໃຫ້ຜູ້ຂຽນໂປຣແກຣມບໍ່ເຮັດວຽກໜັກຄືເມື່ອກ່ອນ ມີແຕ່ຮຽນຮູ້ ແລະ ປະຍຸກໃຊ້ເຄື່ອງມືເຫຼົ່ານັ້ນກໍ່ສາມາດພັດທະນາລະບົບງ່າຍຂຶ້ນ.

### 3.1.5 ໄລຍະທີ 5 ການບໍາລຸງຮັກສາ (Maintenance Phase)

ໂດຍປົກກະຕິແລ້ວໄລຍະການບໍາລຸງຮັກສາຈະບໍ່ນໍາເຂົ້າໄປລວມໃນສ່ວນຂອງ SDLC ຈົນກະທັ້ງລະບົບມີການຕິດຕັ້ງເພື່ອໃຊ້ງານແລ້ວເທົ່ານັ້ນ ໄລຍະນີ້ຈະໃຊ້ເວລາຍາວນານທີ່ສຸດເມື່ອທຽບກັບໄລຍະອື່ນໆທີ່ຜ່ານມາ ເນື່ອງຈາກລະບົບຈະຕ້ອງໄດ້ຮັບການບໍາລຸງຮັກສາຕະຫຼອດໄລຍະເວລາທີ່ມີການໃຊ້ລະບົບສິ່ງທີ່ຄາດຫວັງຂອງໜ່ວຍງານກໍ່ຄືຕ້ອງການໃຫ້ລະບົບໃຊ້ງານຍາວນານຫຼາຍປີລະບົບສາມາດຮອງຮັບເຕັກໂນໂລຊີໃໝ່ໆໃນອະນາຄົດໄດ້ ດັ່ງນັ້ນ ໃນຊ່ວງໄລຍະເວລາດັ່ງກ່າວຈຶ່ງສາມາດເພີ່ມເຕີມຄວາມສາມາດຂອງລະບົບໃຫ້ມີປະສິດທິພາບສູງຂຶ້ນພ້ອມທັງການແກ້ໄຂປັບປຸງໂປຣແກຣມໃນກໍລະນີທີ່ເຫັນຂໍ້ຜິດພາດ.

- ການບໍາລຸງຮັກສາລະບົບ (System Maintenance).
- ການເພີ່ມເຕີມຄວາມສາມາດໃໝ່ໆເຂົ້າໃນລະບົບ (Enhance System).
- ສະໜັບສະໜູນງານຂອງຜູ້ໃຊ້ (Support the User).

ຈາກໄລຍະຕ່າງໆຕາມຂັ້ນຕອນການພັດທະນາລະບົບຕາມແບບແຜນຂອງ SDLC ຈະເຫັນວ່າມີການໃຊ້ຄໍາວ່າໄລຍະ ແລະ ກິດຈະກຳເຊິ່ງສາມາດອະທິບາຍລາຍລະອຽດເພື່ອໃຫ້ເກີດຄວາມເຂົ້າໃຈກົງກັນດັ່ງນີ້:

- ໄລຍະ (Phase) ຄືກຸ່ມກິດຈະກຳທີ່ກ່ຽວຂ້ອງກັນ.
- ກິດຈະກຳ (Activity) ຄືກຸ່ມຂອງງານທີ່ກ່ຽວຂ້ອງກັນ.
- ໜ້າວຽກ (Task) ຄືວຽກທີ່ດຳເນີນການເຊິ່ງຖືເປັນວຽກທີ່ນ້ອຍທີ່ສຸດ.

## 3.2 ເຄື່ອງມືທີ່ໃຊ້ໃນການພັດທະນາ (Development Tools)

### 1) Hardware:

- ເຄື່ອງຄອມພິວເຕີ 1 ໜ່ວຍລຸ້ນ Lenovo Intel(R) Core(TM) I5-3230M
- ເຄື່ອງຄອມພິວເຕີທີ່ໃຊ້ສໍາລັບຂຽນໂປຣແກຣມມີ Spec ດັ່ງນີ້:
  - CPU @2.60GHz,
  - RAM 8GB DDR3L 1600MHz.

- SSD 240 GB, HDD 500 GB.
- Pocket WiFi Modem ເພື່ອໃຊ້ເຊື່ອມຕໍ່ Internet.

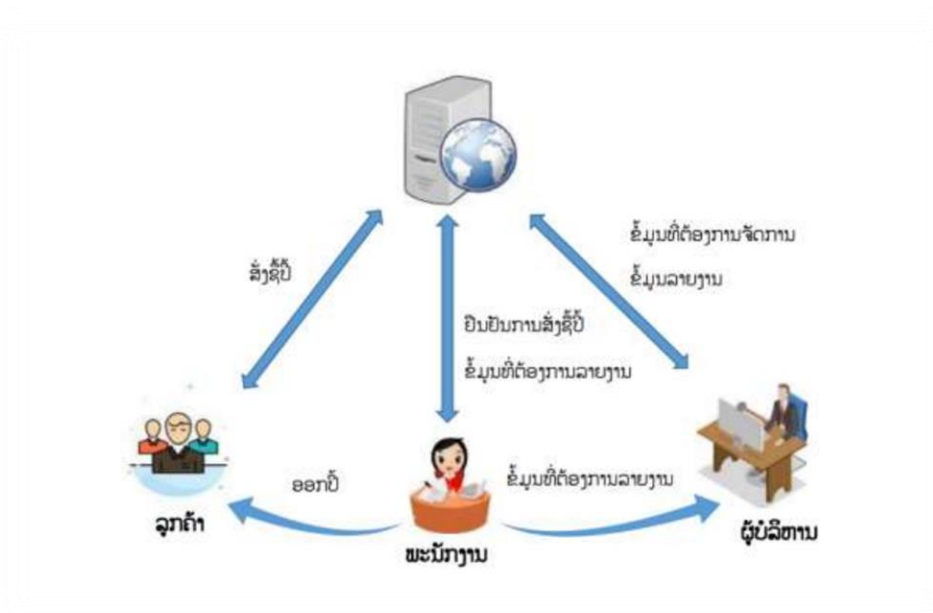
## 2) Software:

- 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.

## 3.3 ການວິເຄາະຂໍ້ມູນ

### 3.3.1 ແຜນວາດລວມຂອງລະບົບ

ລະບົບຂາຍປີ້ລົດເມອອນລາຍຂອງສະຖານນີຂົນສົ່ງໂດຍສານສາຍໃຕ້



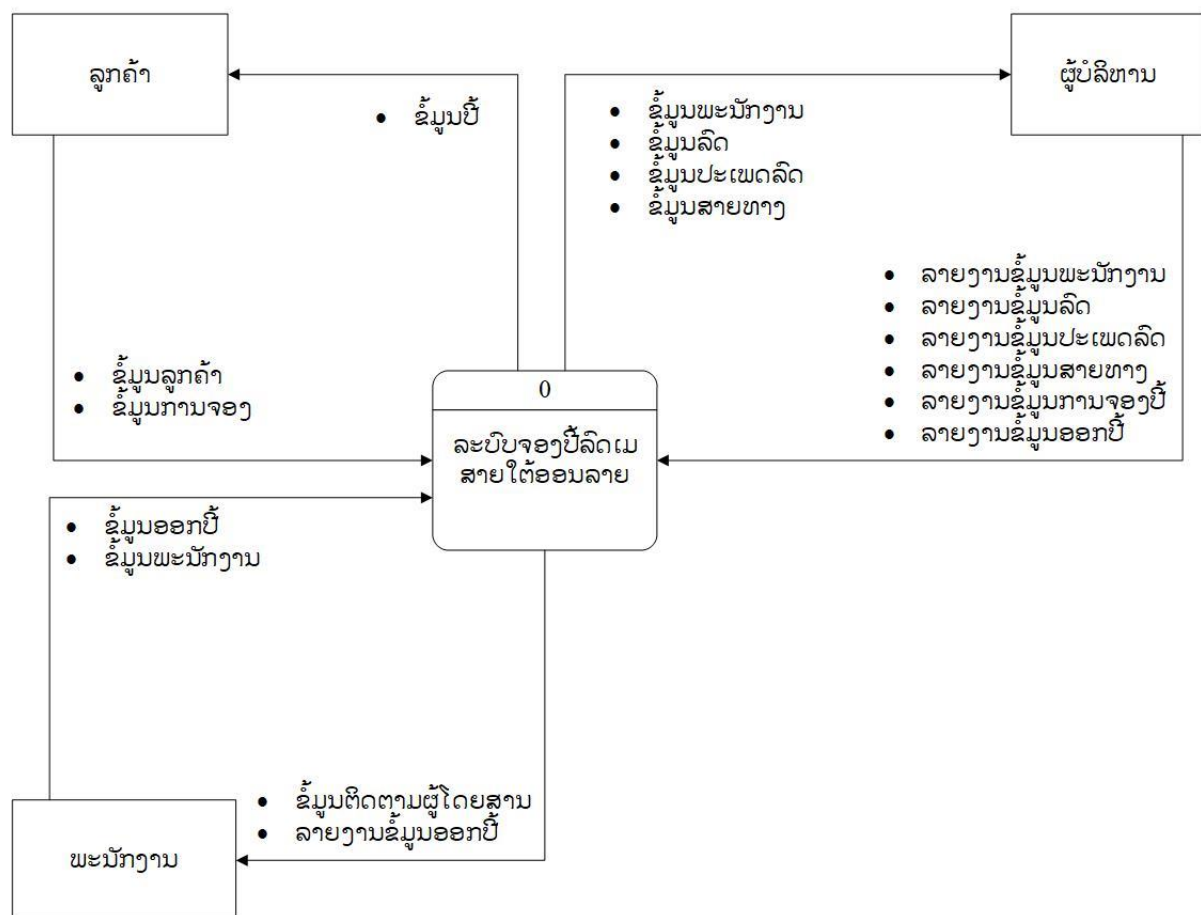
ແຜນວາດທີ 2 : ແຜນວາດວົງຈອນພັດທະນາລະບົບ

### 3.3.2 ຕາຕະລາງສະແດງລາຍລະອຽດຕ່າງໆທີ່ກ່ຽວກັບລະບົບ

External Entity	Process	Data Store
<b>ລູກຄ້າ</b> <b>ພະນັກງານ</b> <b>ຜູ້ບໍລິຫານ</b>	<b>1. ຈັດການຂໍ້ມູນພື້ນຖານ</b> 1.1. ຈັດການຂໍ້ມູນພະນັກງານ 1.2. ຈັດການຂໍ້ມູນບໍລິສັດ 1.3. ຈັດການຂໍ້ມູນປະເພດລົດ 1.4. ຈັດການຂໍ້ມູນລົດ 1.5. ຈັດການຂໍ້ມູນສາຍທາງ <b>2. ລົງທະບຽນ</b> 2.1. ສະມັກສະມາຊິກ 2.2. ເຂົ້າສູ່ລະບົບ <b>3. ບໍລິການ</b> 3.1. ຈອງປີ້ 3.2. ອອກປີ້ <b>4. ລາຍງານ</b> 4.1. ຂໍ້ມູນການຈອງ 4.2. ຂໍ້ມູນພະນັກງານ 4.3. ຂໍ້ມູນສາຍທາງ 4.4. ຂໍ້ມູນລົດ 4.5. ຂໍ້ມູນອອກປີ້	D1 ຂໍ້ມູນພະນັກງານ D2 ຂໍ້ມູນລູກຄ້າ D3 ຂໍ້ມູນບໍລິສັດ D4 ຂໍ້ມູນລົດ D5 ຂໍ້ມູນປະເພດລົດ D6 ຂໍ້ມູນບ່ອນນັ່ງ D7 ຂໍ້ມູນສາຍທາງ D8 ຂໍ້ມູນເວລາລົດອອກ D9 ຂໍ້ມູນການຈອງ D10 ຂໍ້ມູນລາຍລະອຽດການຈອງ D11 ຂໍ້ມູນອອກປີ້ D12 ຂໍ້ມູນຕິດຕາມຜູ້ໂດຍສານ

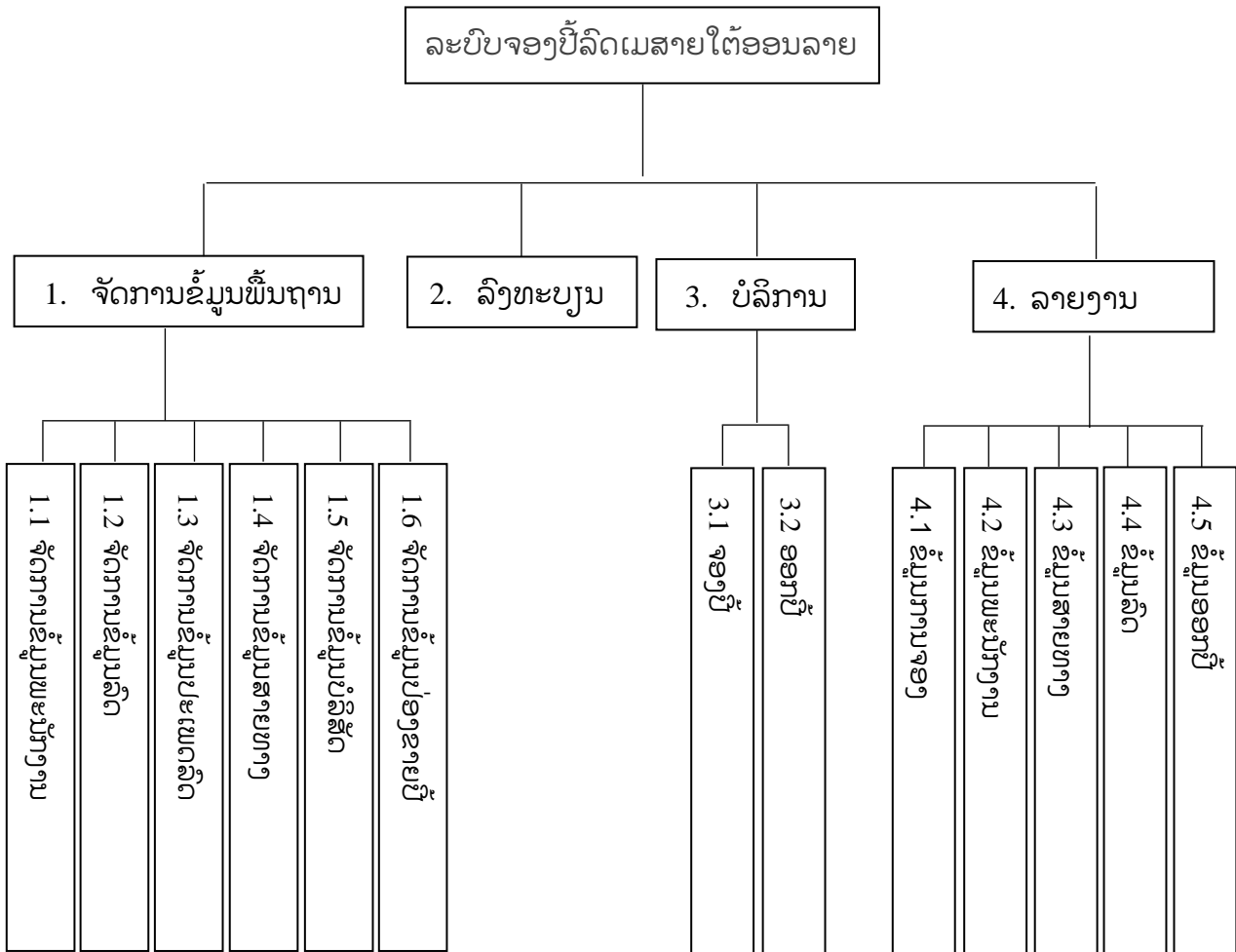
### 3.3.3 ແຜນຈາດເນື້ອຫາ (Context Diagram)

Context Diagram



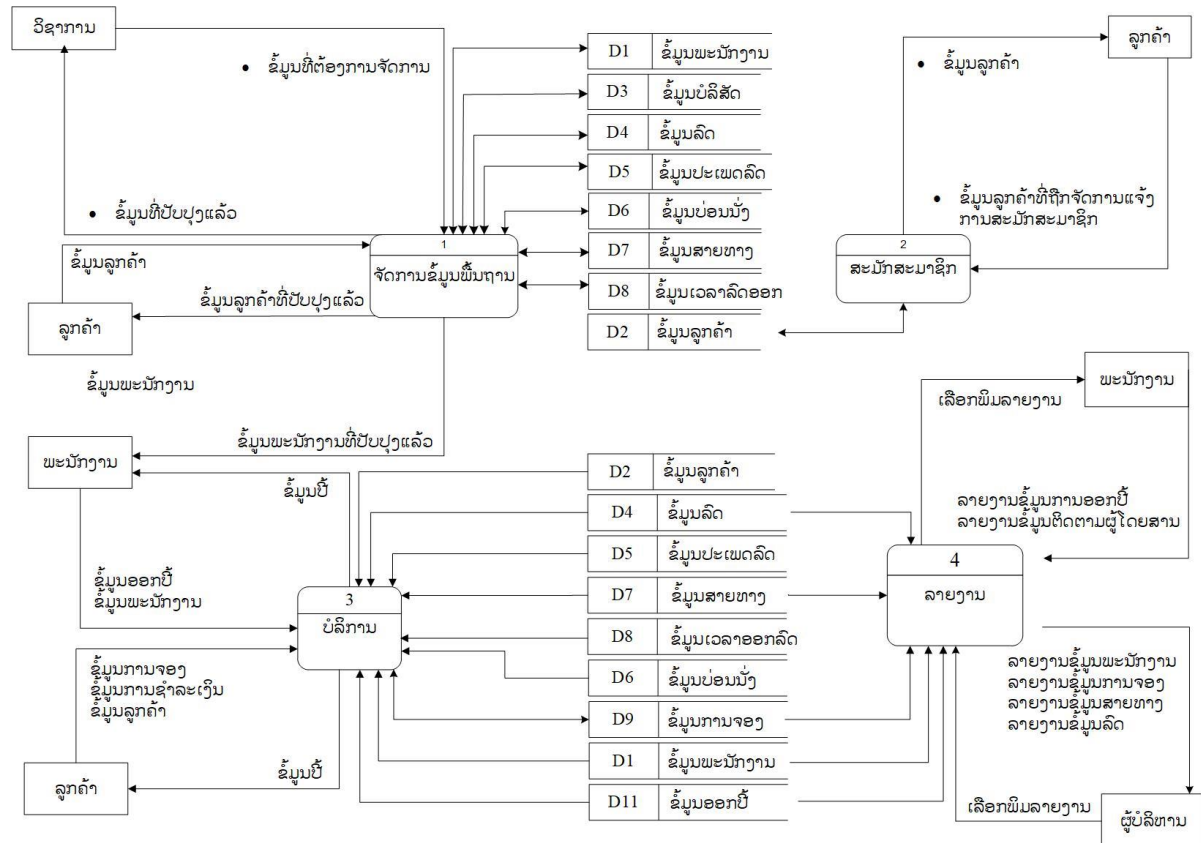


### 3.3.4 ແຜນວາດລຳດັບຊັ້ນໜ້າທີ່ (Functional Hierarchy Diagram)

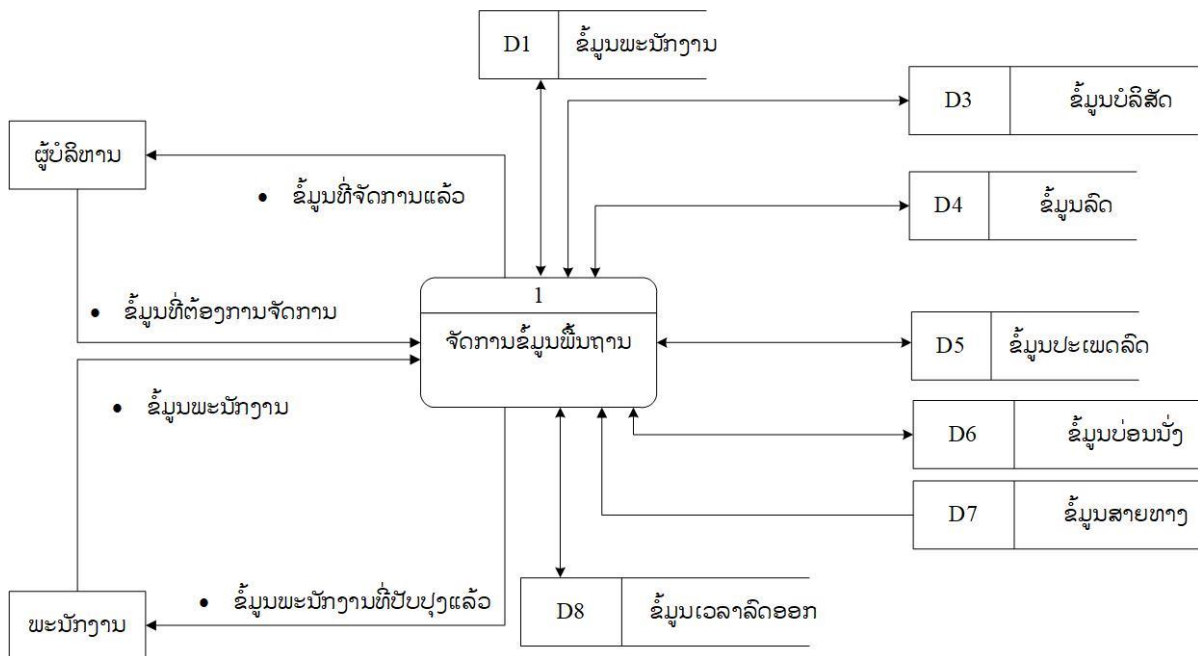


### 3.3.5 ແຜນວາດການໄຫຼຂໍ້ມູນ (Data Flow Diagram: DFD)

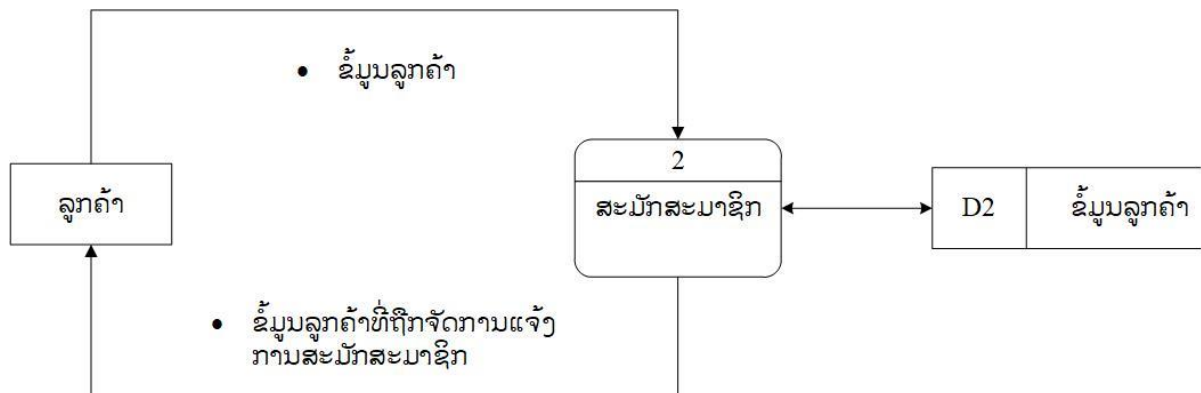
#### 1. ແຜນວາດລວມການໄຫຼຂໍ້ມູນລະດັບ 1 ຂອງແຕ່ລະ Process



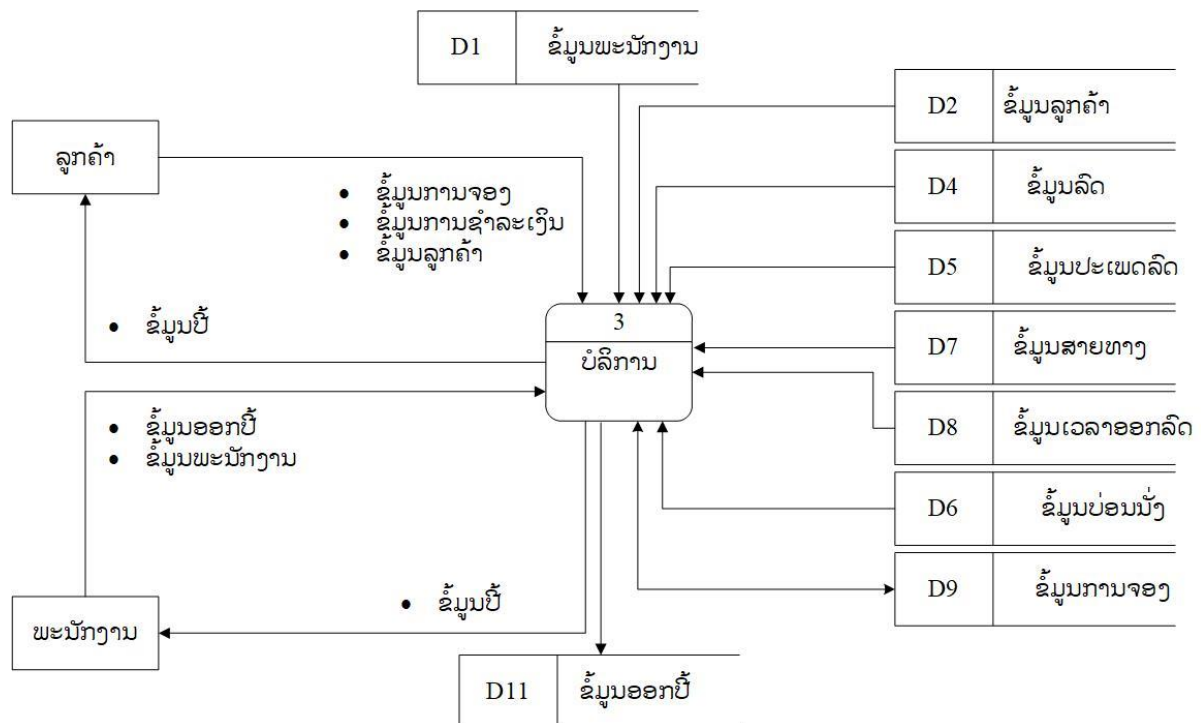
## 2. ແຜນວາດລວມການໄຫຼຂໍ້ມູນລະດັບ 1 ຂອງ Process 1



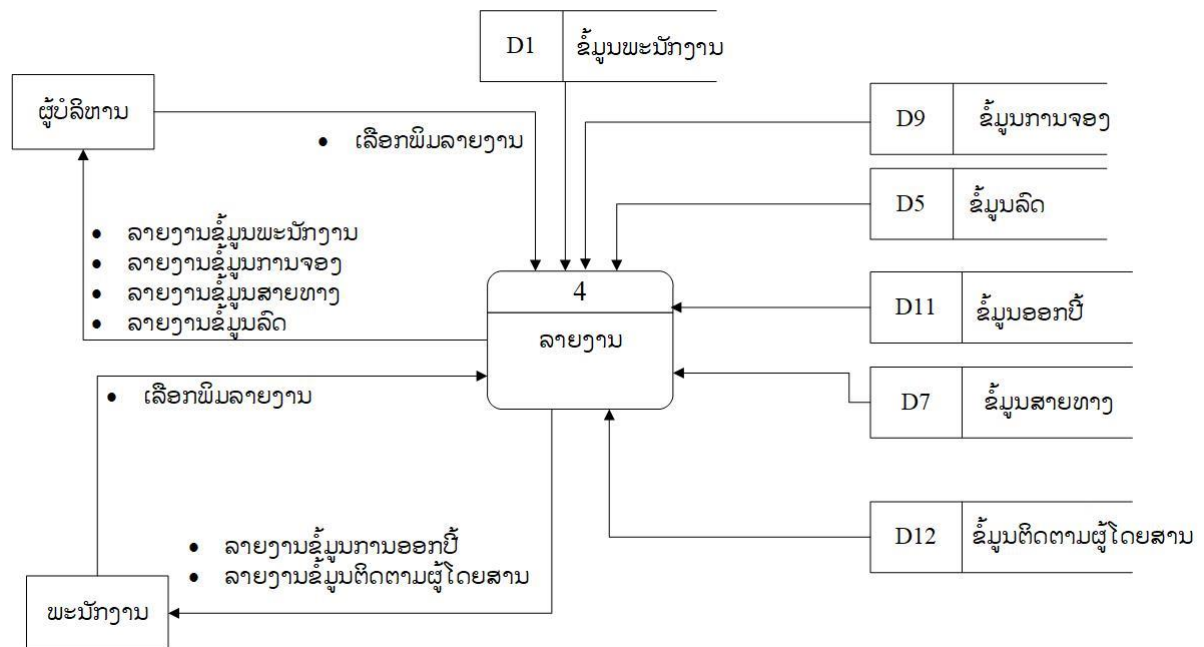
### 3. ແຜນວາດລວມການໄຫຼຂໍ້ມູນລະດັບ 1 ຂອງ Process 2



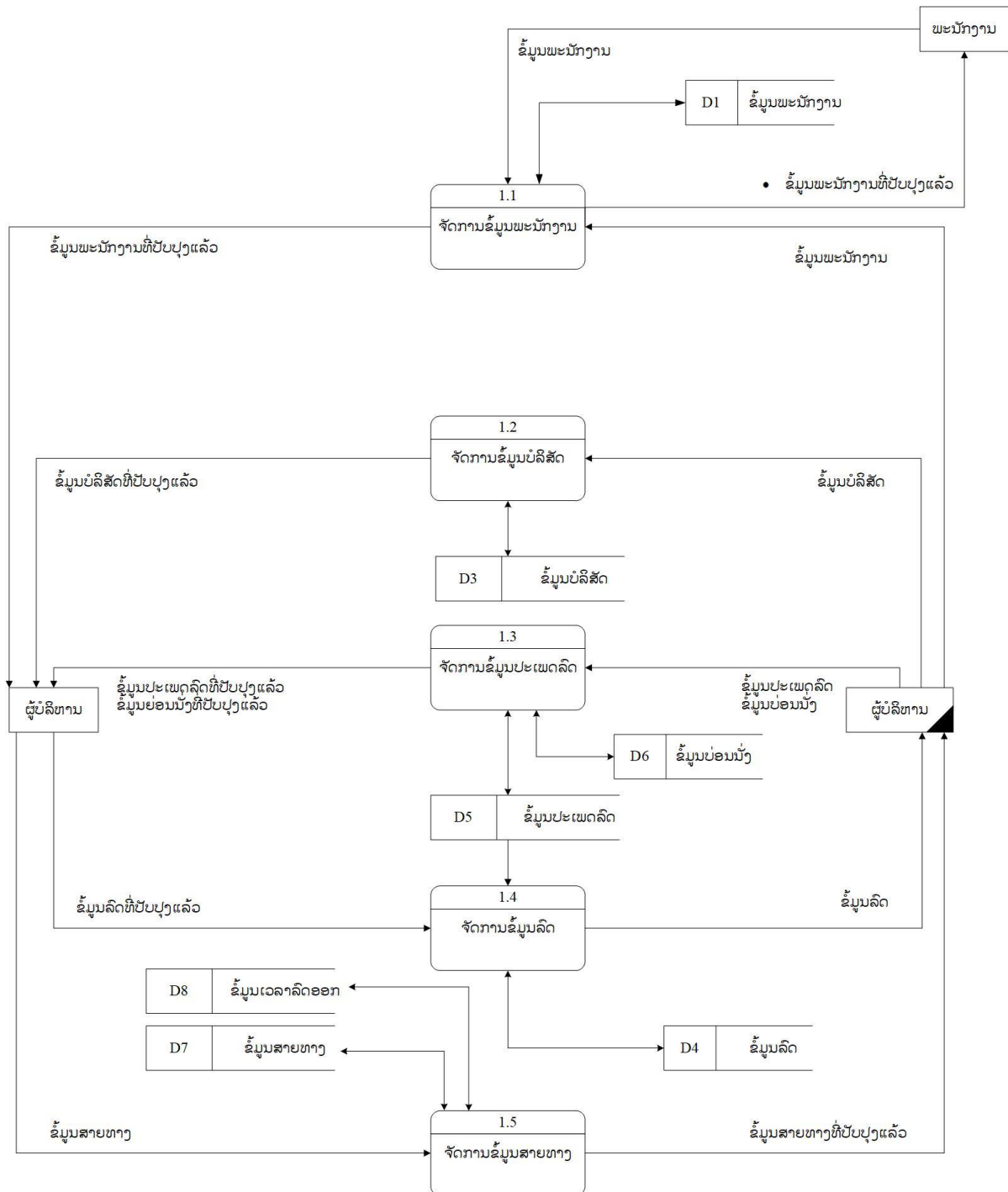
### 4. ແຜນວາດລວມການໄຫຼຂໍ້ມູນລະດັບ 1 ຂອງ Process 3



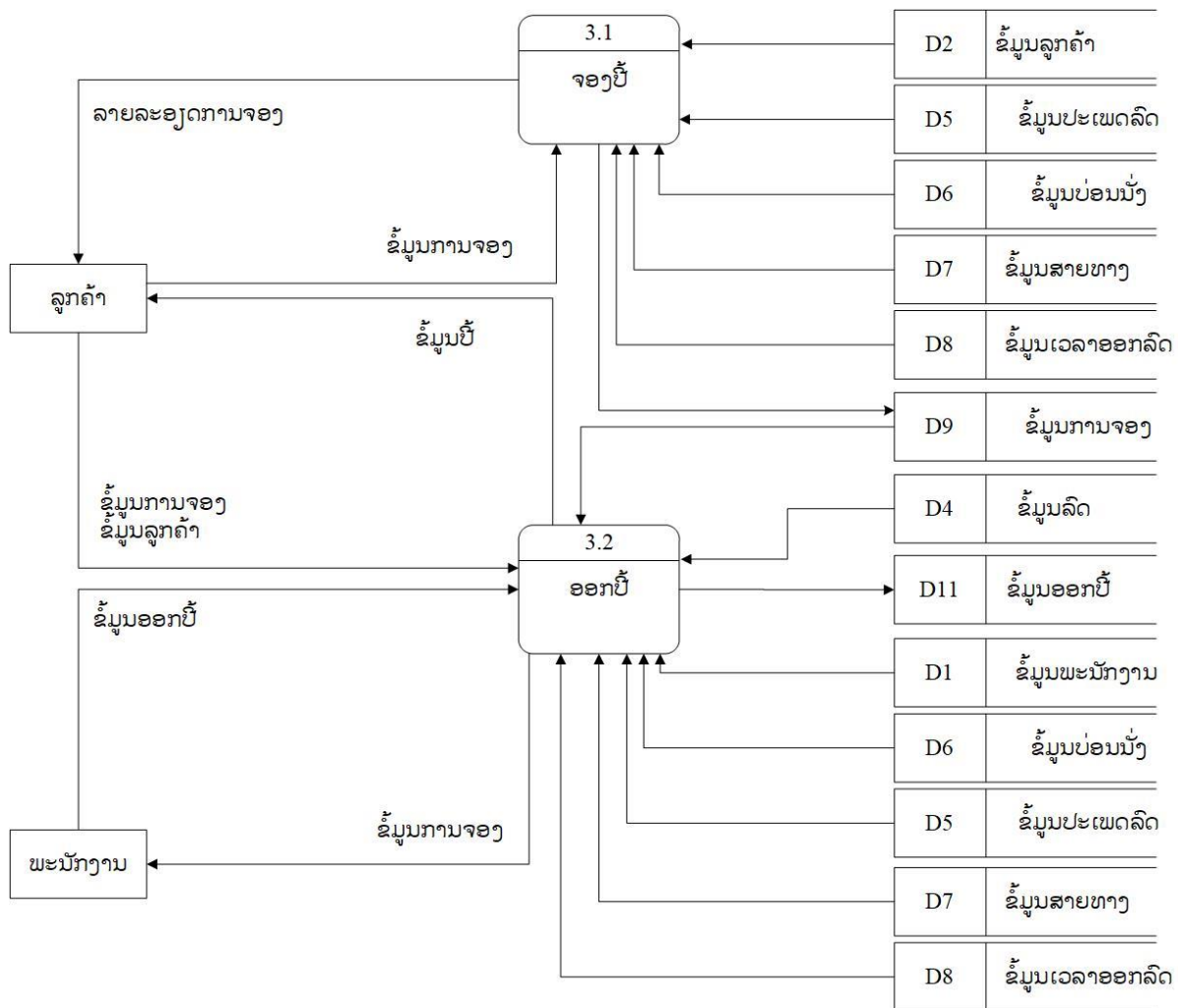
## 5. ແຜນວາດລວມການໄຫຼຂໍ້ມູນລະດັບ 1 ຂອງ Process 4



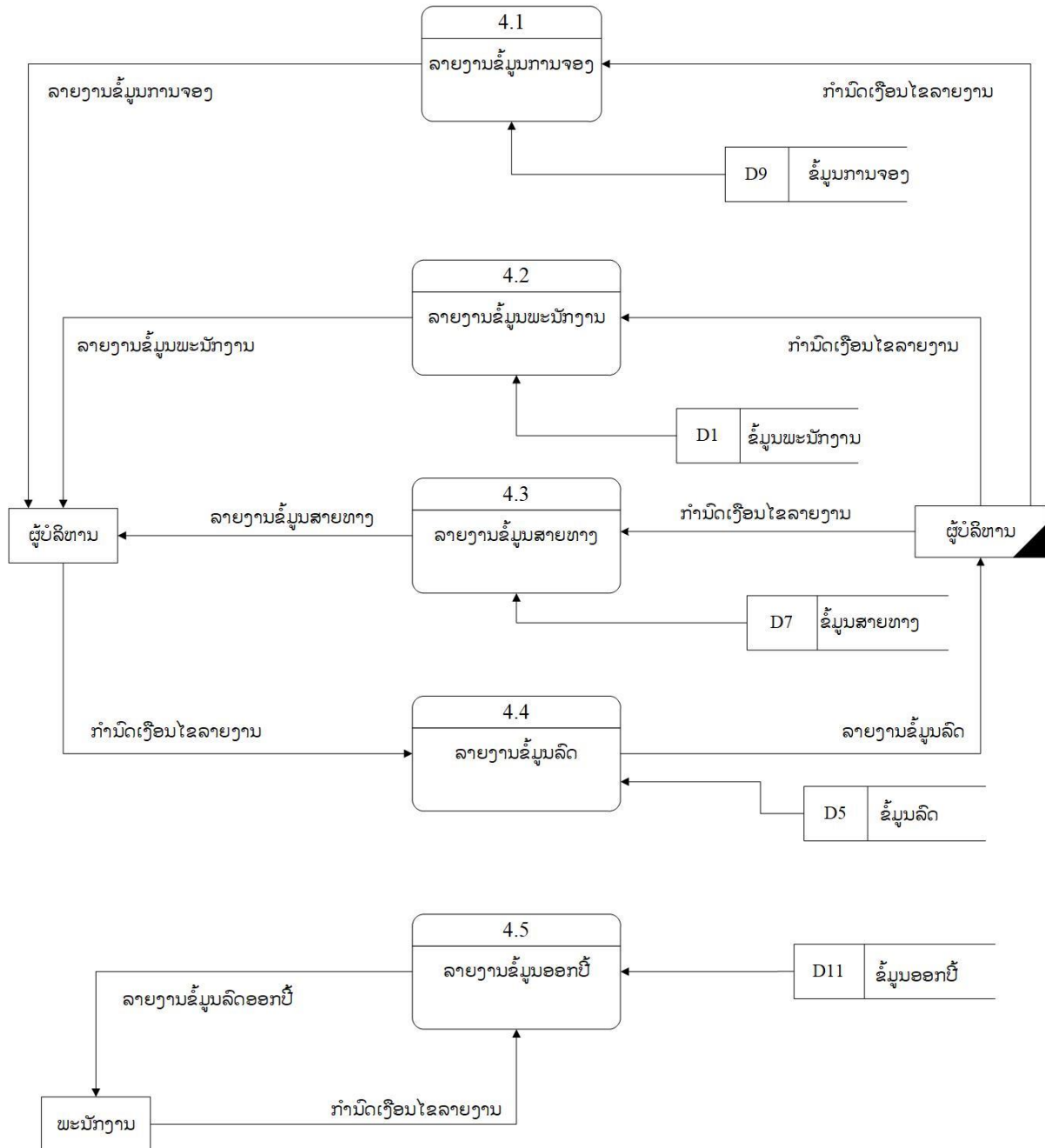
## 6. ແຜນວາດລວມການໄຫຼຂໍ້ມູນລະດັບ 2 ຂອງ Process 1



## 7. ແຜນວາດລວມການໄຫຼຂໍ້ມູນລະດັບ 2 ຂອງ Process 3

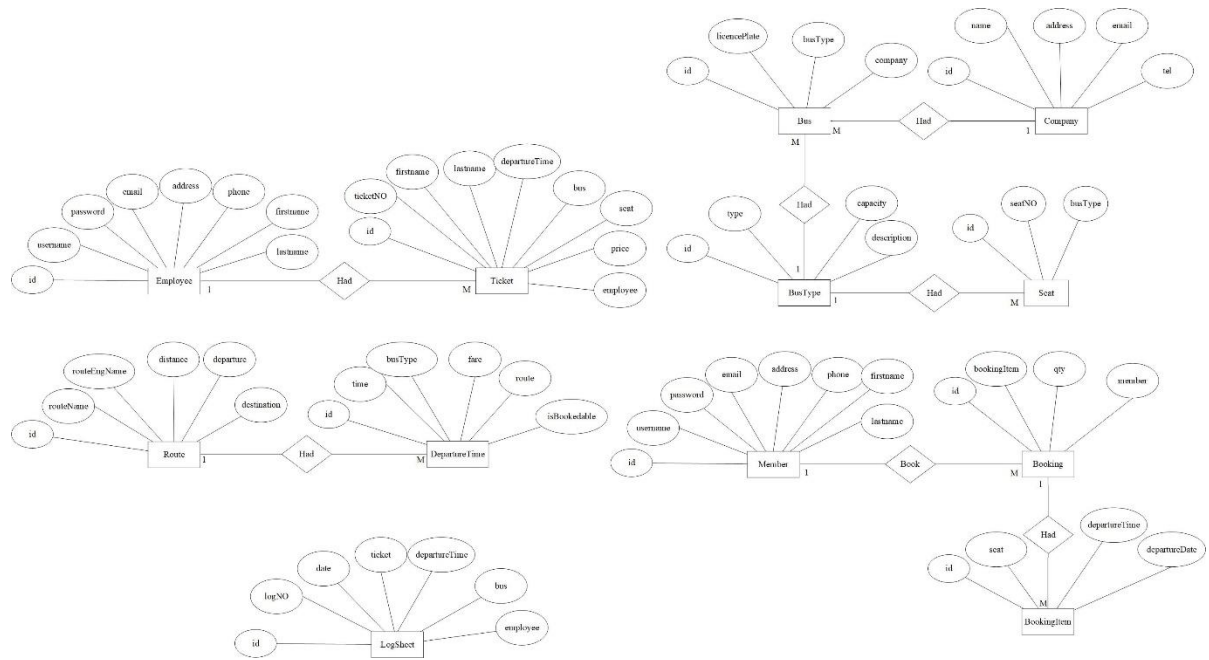


## 8. ແຜນວາດລວມການໄຫຼຂໍ້ມູນລະດັບ 2 ຂອງ Process 4





### 3.3.6 ແຜນວາດຄວາມສຳພັນຂອງຂໍ້ມູນ (ER Diagram)



### 3.4 ການອອກແບບລະບົບ

#### 3.4.1 ການອອກແບບຮ່າງສະແດງຜົນ (Output Design)

#### 3.4.2 ການອອກແບບຮ່າງປ້ອນຂໍ້ມູນ (Input Design)

❖ ຟອມເຂົ້າສູ່ລະບົບແອດມິນ

ຢືນດີຕ້ອນຮັບເຂົ້າສູ່ລະບົບຈັດການ  
ຂໍ້ມູນ! 🖐️

Username  
demo@gogo.com

Password  
.....

ເຂົ້າສູ່ລະບົບ

1

2

ຮູບທີ 1 : ຮູບແບບຟອມເຂົ້າສູ່ລະບົບແອດມິນ

1. ພາກສ່ວນປ້ອນຊື່ຜູ້ໃຊ້ ແລະ ລະຫັດຜ່ານ
2. ເຂົ້າສູ່ລະບົບ

❖ ຟອມການຈັດການຂໍ້ມູນບໍລິສັດ

The screenshot shows a web interface for managing companies. At the top right, there is a green button with a plus sign and the text '+ ເພີ່ມ' (Add), labeled with callout 1. Below this is a table with the title 'ຈັດການຂໍ້ມູນບໍລິສັດ' (Manage Company Information). The table has columns for '#', 'ບໍລິສັດ' (Company), 'ທີ່ຢູ່' (Address), 'ອີເມວ' (Email), 'ເບີໂທ' (Phone), and 'ACTIONS'. There are three rows of data. The 'ACTIONS' column contains two icons: a green checkmark and a red trash can. Callout 2 points to the green checkmark icon in the first row. Callout 3 points to the red trash can icon in the third row.

#	ບໍລິສັດ	ທີ່ຢູ່	ອີເມວ	ເບີໂທ	ACTIONS
1	ບໍລິສັດດຽງໂກ	ນະຄອນຫຼວງ	company@gmail.com	030 9873845	
2	ບໍລິສັດຈິດປະສົງ	ມ ໄຊທາ	company@gmail.com	030 9873845	
3	ແກ້ໄຂທ	ຄຳຮຸກເກີກ	phonekham.dev@gmail.com	020 28022677	

ຮູບທີ 2 : ຮູບແບບຟອມການຈັດການຂໍ້ມູນບໍລິສັດ

1. ການເພີ່ມ
2. ແກ້ໄຂ
3. ການລຶບ

❖ ຟອມການເພີ່ມຂໍ້ມູນບໍລິສັດ

The diagram shows a web form titled "ເພີ່ມຂໍ້ມູນບໍລິສັດ" (Add Company Information). The form contains four input fields: "ບໍລິສັດ" (Company), "ທີ່ຢູ່" (Address), "ອີເມວ" (Email), and "ເບີໂທ" (Phone). Below the form are two buttons: "ບັນທຶກ" (Save) and "ຍົກເລີກ" (Cancel). Numbered callouts are as follows: Callout 1 points to the form title; Callout 2 points to the "ບັນທຶກ" button; Callout 3 points to the "ຍົກເລີກ" button.

ຮູບທີ 3 : ຮູບແບບຟອມການເພີ່ມຂໍ້ມູນບໍລິສັດ

1. ພາກສ່ວນປ້ອນຂໍ້ມູນ
2. ບັນທຶກ
3. ຍົກເລີກ

❖ ຟອມການຈັດການຂໍ້ມູນສາຍທາງ

1

2

3

4

ສາຍທາງ	ສາຍທາງພາສາອັງກິດ	ໄລຍະທາງ KM	ຕົ້ນທາງ	ປາຍທາງ	ຈັດການ
ວຽງຈັນ - ຫຼີກ 20	VTE - LAK 20	110	ວຽງຈັນ	ຫຼີກ 20	ຈັດການ
ວຽງຈັນ - ປາກເຊ	VTE-PAKSE	120	ວຽງຈັນ	ປາກເຊ	ຈັດການ

ຮູບທີ 4 : ຮູບແບບຟອມການຈັດການຂໍ້ມູນສາຍທາງ

4. ການເພີ່ມ
5. ການຄົ້ນຫາ
6. ການແກ້ໄຂ
7. ການລຶບ

❖ ຟອມການເພີ່ມຂໍ້ມູນສາຍທາງ

1

ເພີ່ມຂໍ້ມູນສາຍທາງໃໝ່

ປາຍທາງ

ໄລຍະທາງ

ສາຍທາງພາສາອັງກິດ

ສາຍທາງ ວຽງຈັນ -

ບັນທຶກ

ຍົກເລີກ

2

3

ຮູບທີ 5 : ຮູບແບບຟອມການເພີ່ມຂໍ້ມູນສາຍທາງ

1. ພາກສ່ວນການປ້ອນຂໍ້ມູນ
2. ບັນທຶກ
3. ຍົກເລີກ

❖ ຟອມການຈັດການຖ້ວງລົດ

1

ຈັດການຂໍ້ມູນຖ້ວງລົດປະຈຳສາຍ



ວຽງຈັນ - ຫຼັກ 20

VTE - LAK 20

2

3

+ ເພີ່ມຖ້ວງໃໝ່

#	ເວລາອອກ	ຄ່າໂດຍສານ	ປະເພດລົດ	ສາມາດຈອງຝ່າຍເວັບ	ຈັດການ
1	8:30	100000	ທຳມະດາ	✗	 

ຮູບທີ 6 : ຮູບແບບຟອມການຈັດການຖ້ວງລົດ

1. ການເພີ່ມຖ້ວງໃໝ່
2. ການແກ້ໄຂ
3. ການລຶບ

❖ ຟອມການເພີ່ມຂໍ້ມູນຖ້ວນລົດ

The image shows a web form titled "ເພີ່ມຂໍ້ມູນທໍຽວໂດຍສານ" (Add vehicle information). The form contains the following fields and controls:

- 1**: Points to the title bar of the form window.
- 2**: Points to a dropdown menu labeled "ເລືອກປະເພດລົດ" (Select vehicle type).
- 3**: Points to a checkbox labeled "ຈອງຜ່ານເວັບ" (Register via website).
- 4**: Points to a blue button labeled "ບັນທຶກ" (Save/Record).

Other visible fields include "ເວລາອອກ" (Departure time) and "ຄ່າໂດຍສານ" (Vehicle fee).

ຮູບທີ 7 : ຮູບແບບຟອມການເພີ່ມຂໍ້ມູນຖ້ວນລົດ

1. ພາກສ່ວນປ້ອນຂໍ້ມູນຖ້ວນໂດຍສານ
2. ເລືອກປະເພດລົດໂດຍສານ
3. ເລືອກຊ່ອງທາງການສັ່ງຈອງ
4. ບັນທຶກການເພີ່ມຂໍ້ມູນຖ້ວນລົດ



❖ ຟອມການຈັດການປະເພດລົດ

The screenshot shows a web application for managing vehicle types. At the top, there is a title 'ຈັດການຂໍ້ມູນປະເພດລົດ' (Manage Vehicle Type Information). Below the title is a table with 6 columns: '#', 'ປະເພດ' (Type), 'ລາຍນະຄອນ' (City/Region), 'ປະເພດບ່ອນນັ່ງ' (Seating Type), 'ຈຳນວນບ່ອນນັ່ງ' (Number of Seats), and 'ຈັດການບ່ອນນັ່ງ' (Manage Seating). The table contains 4 rows of data. To the left of the table is a green button with a plus sign and the text '+ ເພີ່ມ' (Add). To the right of the table are two columns of action buttons: 'ຈັດການ' (Manage) and 'ຈັດການ' (Manage). Callout 1 points to the '+ ເພີ່ມ' button. Callout 2 points to the 'ຈັດການ' button in the first row. Callout 3 points to the 'ຈັດການ' button in the first row. Callout 4 points to the 'ຈັດການ' button in the first row.

#	ປະເພດ	ລາຍນະຄອນ	ປະເພດບ່ອນນັ່ງ	ຈຳນວນບ່ອນນັ່ງ	ຈັດການບ່ອນນັ່ງ	ຈັດການ
1	ທຳມະດາ	ລົດທຳມະດາ 45 ບ່ອນນັ່ງ	ຕຽງນັ່ງຊັ້ນຕຽວ	45	ຈັດການ	ຈັດການ
2	VIP	ລົດ VIP 45 ບ່ອນນອນ	ຕຽງນອນສອງຊັ້ນ	45	ຈັດການ	ຈັດການ
3	ລົດດ່ວນ2	ລົດຕຽງນັ່ງ ທຳມະດາ 45 ບ່ອນນັ່ງ	ຕຽງນອນສອງຊັ້ນ	45	ຈັດການ	ຈັດການ
4	VIP 2	ລົດຕຽງນອນ 45 ບ່ອນນັ່ງ	ຕຽງນອນສອງຊັ້ນ	45	ຈັດການ	ຈັດການ

ຮູບທີ 8 : ຮູບແບບຟອມການຈັດການປະເພດລົດ

1. ການເພີ່ມ
2. ການຈັດການປະເພດບ່ອນນັ່ງ
3. ການແກ້ໄຂ
4. ການລຶບ

❖ ຟອມການເພີ່ມປະເພດລົດ

The image shows a web form titled "ຟອມເພີ່ມປະເພດລົດ" (Vehicle Type Addition Form). It contains several input fields and buttons. Numbered callouts are as follows:

- 1:** Points to the main form container.
- 2:** Points to the "ປະເພດປ່ອນນັ່ງ" (Vehicle Type) section, which includes radio buttons for "ດັ່ງຊັ້ນດຽວ" (Single Class) and "ຕຽງສອງຊັ້ນ" (Two Classes).
- 3:** Points to the "ບັນທຶກ" (Save) button.
- 4:** Points to the "ຍົກເລີກ" (Delete) button.

Other visible fields include "ປະເພດ" (Type), "ລາຍລະອຽດ" (Details), and "ຈຳນວນປ່ອນນັ່ງ" (Number of Vehicles).

ຮູບທີ 9 : ຮູບແບບຟອມການເພີ່ມປະເພດລົດ

1. ພາກສ່ວນປ້ອນຂໍ້ມູນການເພີ່ມປະເພດລົດ
2. ເລືອກປະເພດປ່ອນນັ່ງ
3. ບັນທຶກ
4. ຍົກເລີກ

### 3.4.3 ການອອກແບບຖານຂໍ້ມູນ

#### 1. ຕາຕະລາງພະນັກງານ (Employee)

Table: Employee					
Field Name	Data Type	Allow Null	Key	Description	Reference
id	ObjectId	No	PK	ລະຫັດພະນັກງານ	
firstname	String	No		ຊື່ພະນັກງານ	
lastname	String	No		ນາມສະກຸນ	
username	String	No		ຊື່ຜູ້ໃຊ້ພະນັກງານ	
password	String	No		ລະຫັດຜ່ານ	
email	String	No		ອີເມວ	
address	String	No		ທີ່ຢູ່	
phone	String	No		ເບີໂທ	
role	String	No		ໜ້າທີ່	
status	String	No		ສະຖານະ	

#### 2. ຕາຕະລາງຂໍ້ມູນປີ້ (Ticket)

Table: Ticket					
Field Name	Data Type	Allow Null	Key	Description	Reference
id	ObjectId	No	PK	ລະຫັດປີ້	
ticketNo	Number	No		ເລກທີປີ້	
firstname	String	No		ຊື່	
lastname	String	No		ນາມສະກຸນ	

departureTime	ObjectId	No	FK	ເວລາລົດອອກ	DepartureTime
bus	ObjectId	No		ລະຫັດລົດ	
Seat	ObjectId	No		ລະຫັດບ່ອນນັ່ງ	Seat
price	Number	No		ລາຄາປີ້	
employee	ObjectId	No	FK	ລະຫັດຜູ້ອອກປີ້	Employee

### 3. ຕາຕະລາງຂໍ້ມູນບໍລິສັດ (Company)

Table: Company					
Field Name	Data Type	Allow Null	Key	Description	Reference
id	ObjectId	No	PK	ລະຫັດລູກຄ້າ	
name	String	No		ຊື່	
address	String	No		ທີ່ຢູ່	
tel	String	No		ເບີໂທ	
email	String	No		ອີເມວ	

### 4. ຕາຕະລາງຂໍ້ມູນປະເພດລົດ (BusType)

Table: Bus Type					
Field Name	Data Type	Allow Null	Key	Description	Reference
id	ObjectId	No	PK	ລະຫັດລູກຄ້າ	
type	String	No		ປະເພດ	
description	String	No		ລາຍລະອຽດ	

capacity	Number	No		ຈຳນວນບ່ອນນັ່ງ	
floorType	Number	No		ປະເພດບ່ອນນັ່ງ	

#### 5. ຕາຕະລາງຂໍ້ມູນລົດ (Bus)

Table: Bus					
Field Name	Data Type	Allow Null	Key	Description	Reference
id	ObjectId	No	PK	ລະຫັດລົດ	
licencePlate	String	No		ທະບຽນລົດ	
busType	ObjectId	No	FK	ລະຫັດປະເພດລົດ	BusType
company	ObjectId	No	FK	ລະຫັດບໍລິສັດ	Company

#### 6. ຕາຕະລາງຂໍ້ມູນບ່ອນນັ່ງ (Seat)

Table: Seat					
Field Name	Data Type	Allow Null	Key	Description	Reference
id	ObjectId	No	PK	ລະຫັດບ່ອນນັ່ງ	
seatNo	String	No		ເບີບ່ອນນັ່ງ	
busType	ObjectId	No	FK	ລະຫັດປະເພດລົດ	BusType
floor	Number	No		ຊັ້ນບ່ອນນັ່ງ	

7. ຕາຕະລາງຂໍ້ມູນສະມາຊິກ (Member)

Table: Member					
Field Name	Data Type	Allow Null	Key	Description	Reference
id	ObjectId	No	PK	ລະຫັດສະມາຊິກ	
firstname	String	No		ຊື່	
lastname	String	No		ນາມສະກຸນ	
phone	String	No		ເບີໂທ	
username	String	No		ຊື່ຜູ້ໃຊ້	
password	String	No		ລະຫັດຜ່ານ	
email	String	No		ອີເມວ	

8. ຕາຕະລາງຂໍ້ມູນການຈອງ (Booking)

Table: Booking					
Field Name	Data Type	Allow Null	Key	Description	Reference
id	ObjectId	No	PK	ລະຫັດການຈອງ	
bookingItem	ObjectId	No		ລາຍລະອຽດການ ຈອງ	
qty	Number	No		ຈະນວນປີ້	
member	ObjectId	No	FK	ລະຫັດສະມາຊິກ	Member
fullname	String	No		ຊື່ເຕັມ	
tel	String	No		ເບີໂທ	
email	String	No		ອີເມວ	

totalAmount	Number	No		ເປັນເງິນ	
-------------	--------	----	--	----------	--

9. ຕາຕະລາງຂໍ້ມູນລາຍລະອຽດການຈອງ (BookingItem)

Table: BookingItem					
Field Name	Data Type	Allow Null	Key	Description	Reference
id	ObjectId	No	PK	ລະຫັດລາຍລະອຽດການຈອງ	
seat [ ]	ObjectId	No	FK	ລະຫັດບ່ອນນັ່ງ	Seat
departureTime	ObjectId	No	FK	ເວລາລົດອອກ	DepartureTime
departureDate	Date	No		ວັນທີເດີນທາງ	

10. ຕາຕະລາງຂໍ້ມູນສາຍທາງ (Route)

Table: Route					
Field Name	Data Type	Allow Null	Key	Description	Reference
id	ObjectId	No	PK	ລະຫັດສາຍທາງ	
routeName	String	No		ຊື່ສາຍທາງ	
routeEngName	String	No		ຊື່ສາຍທາງອັງກິດ	
distance	Number	No		ໄລຍະທາງ	
departure	String	No		ຕົ້ນທາງ	
destination	String	No		ປາຍທາງ	

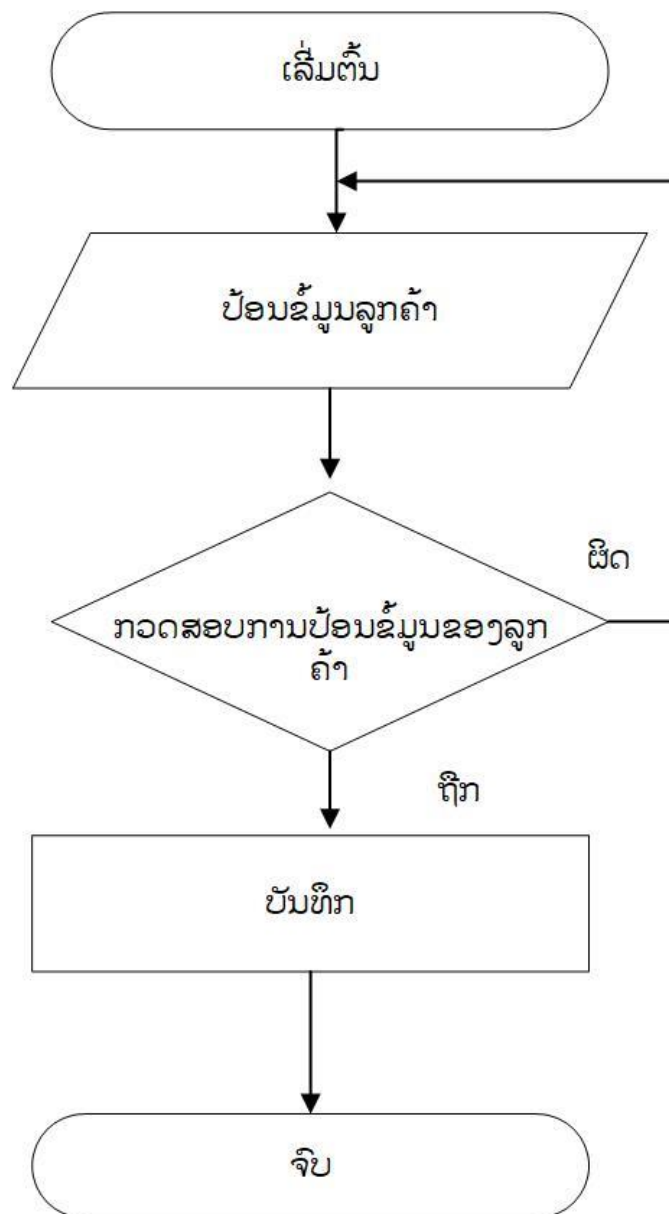
11. ຕາຕະລາງຂໍ້ມູນເວລາລົດອອກ (DepartureTime)

<b>Table: DepartureTime</b>					
Field Name	Data Type	Allow Null	Key	Description	Reference
id	ObjectId	No	PK	ລະຫັດເລລວລາລົດ ອອກ	
time	String	No		ເວລາ	
busType	ObjectId	No	FK	ລະຫັດປະເພດລົດ	BusType
fare	Number	No		ຄ່າໂດຍສານ	
route	ObjectId	No	FK	ລະຫັດສາຍທາງ	Route
isBookable	Boolean	No		ສາມາດຈອງຜ່ານເວັບ	

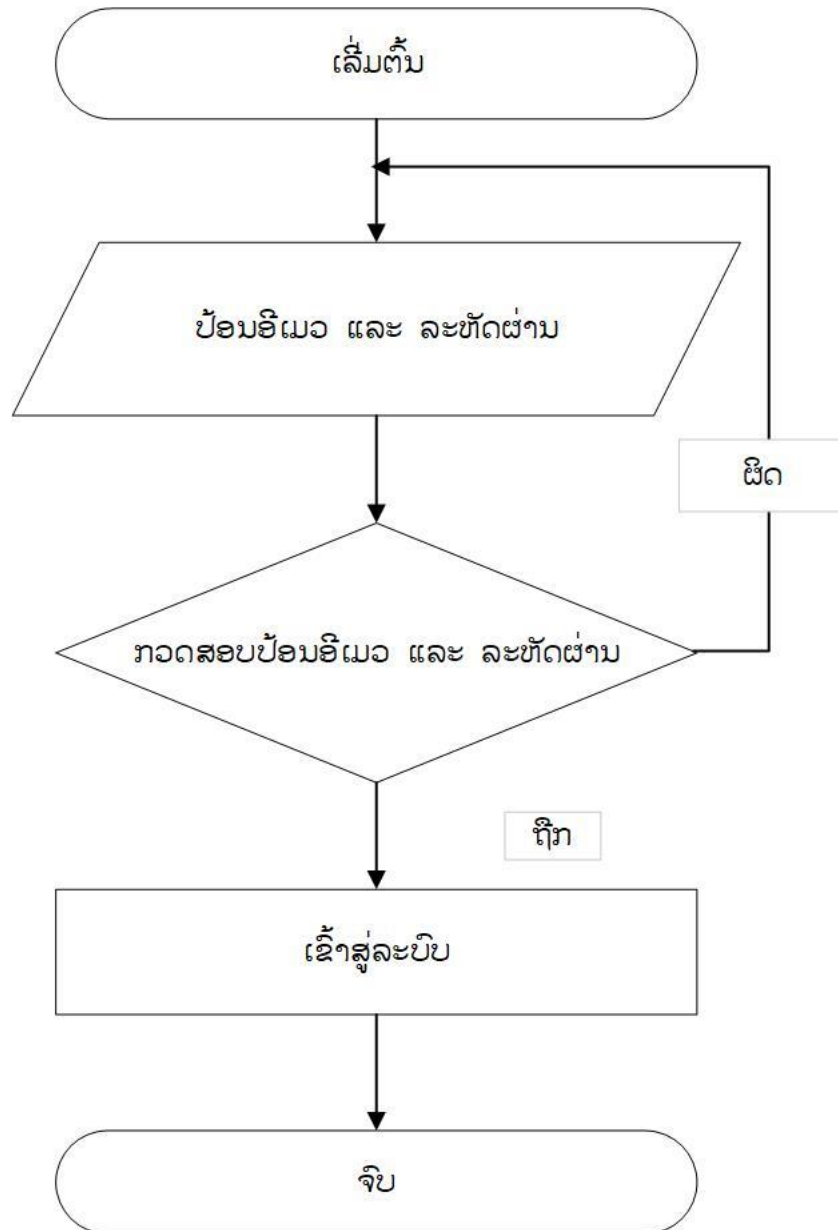


### 3.5 ແຜນວາດຂັ້ນຕອນການເຮັດວຽກ (Flowchart)

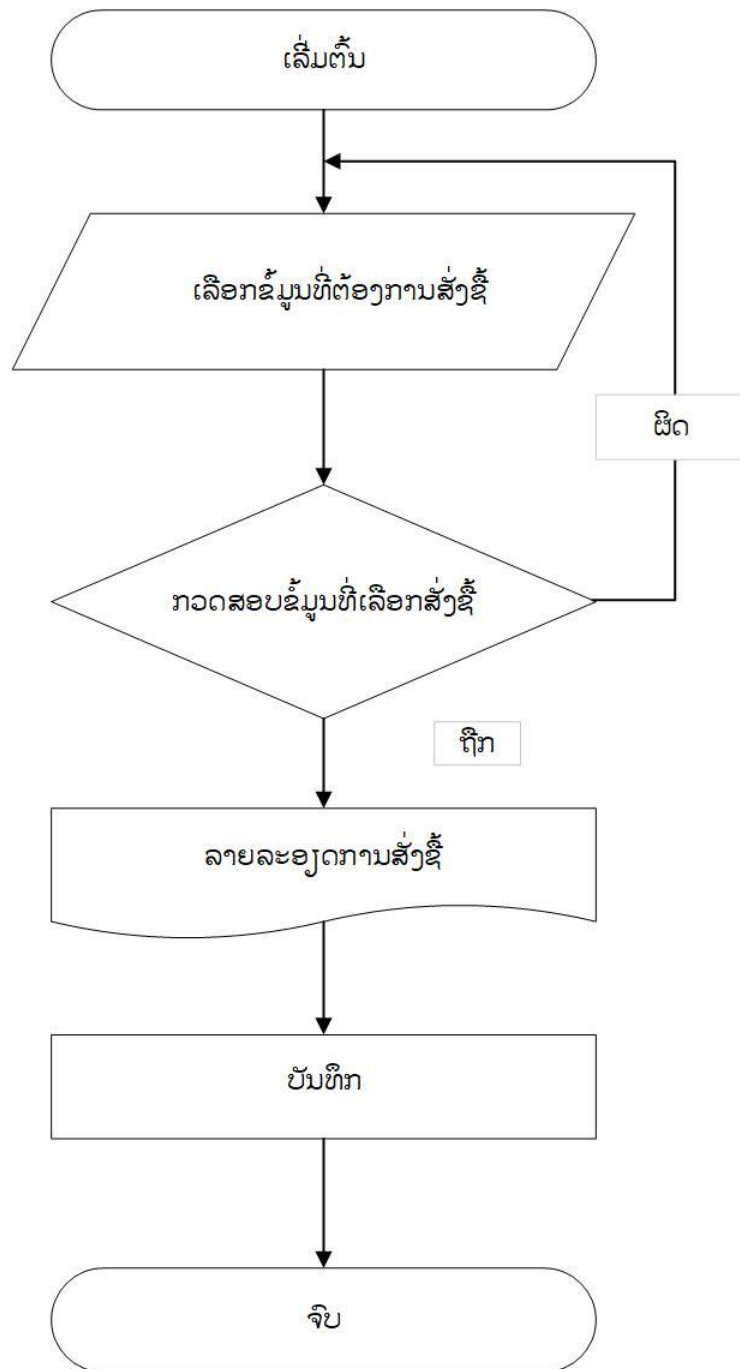
#### 3.5.1 ແຜນວາດ Flowchart ການສະໝັກສະມາຊິກ



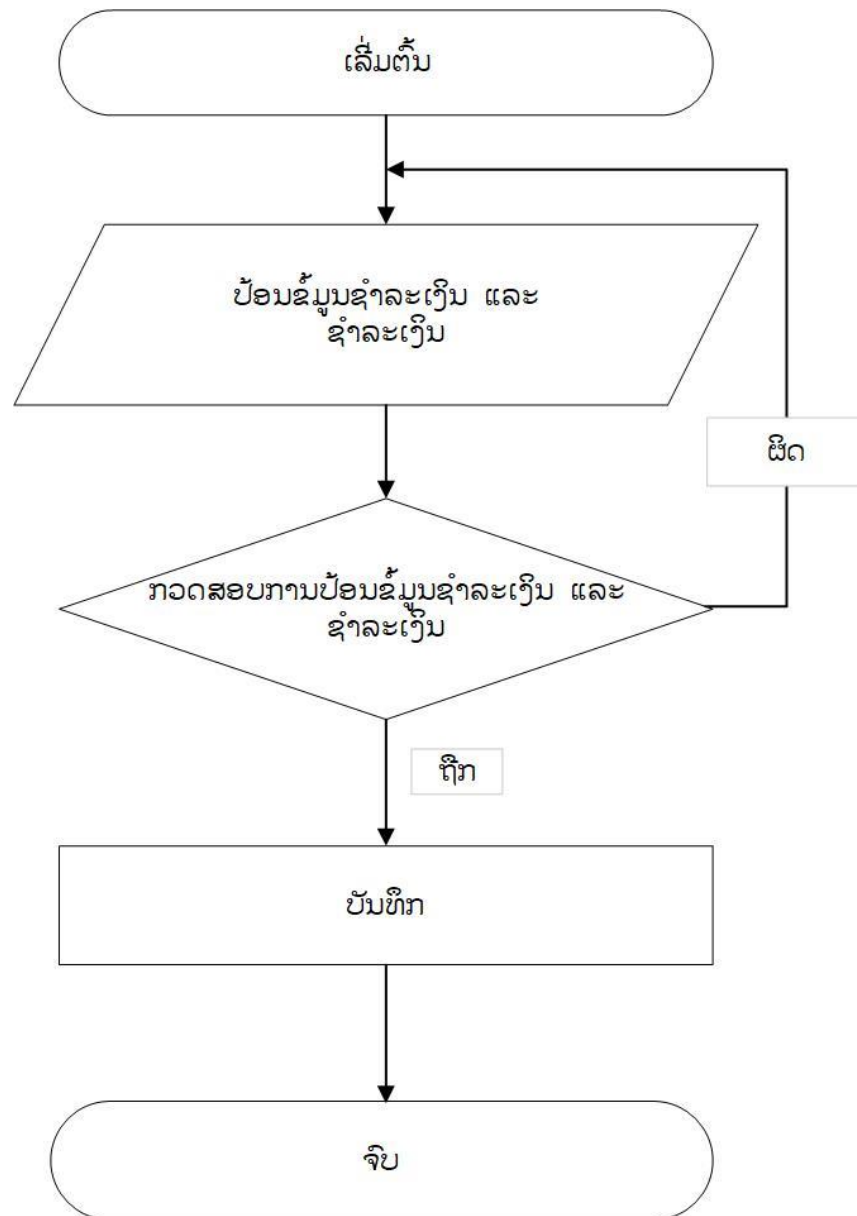
#### 3.5.2 ແຜນວາດ Flowchart ການເຂົ້າສູ່ລະບົບ



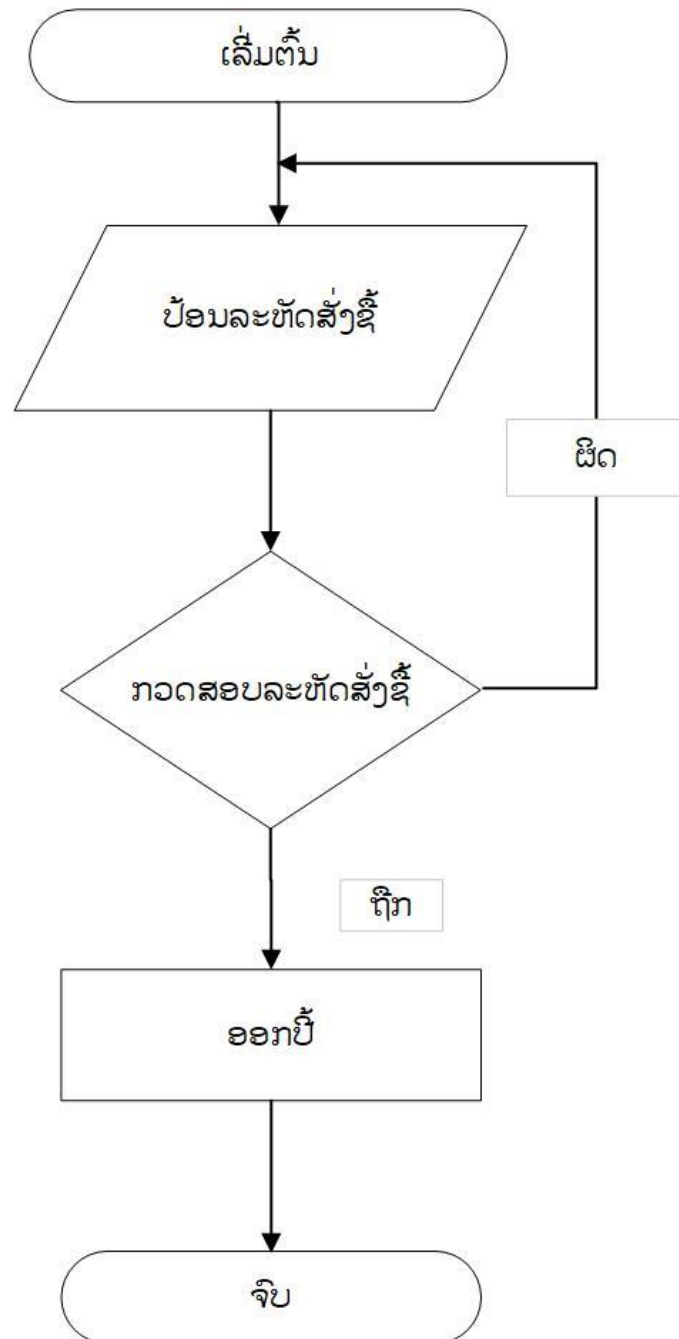
### 3.5.3 ແຜນວາດ Flowchart ການສັ່ງຊື້



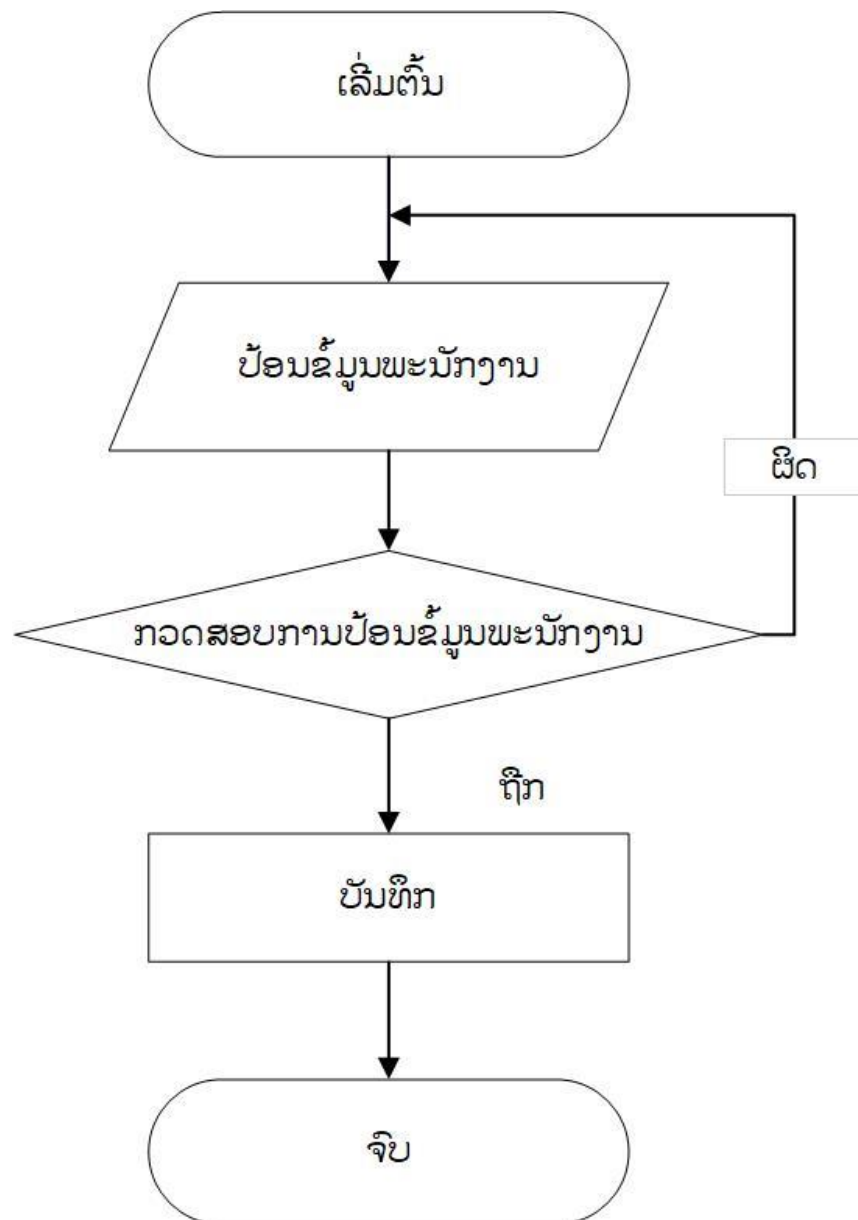
### 3.5.4 ແຜນວາດ Flowchart ການຊຳລະເງິນ



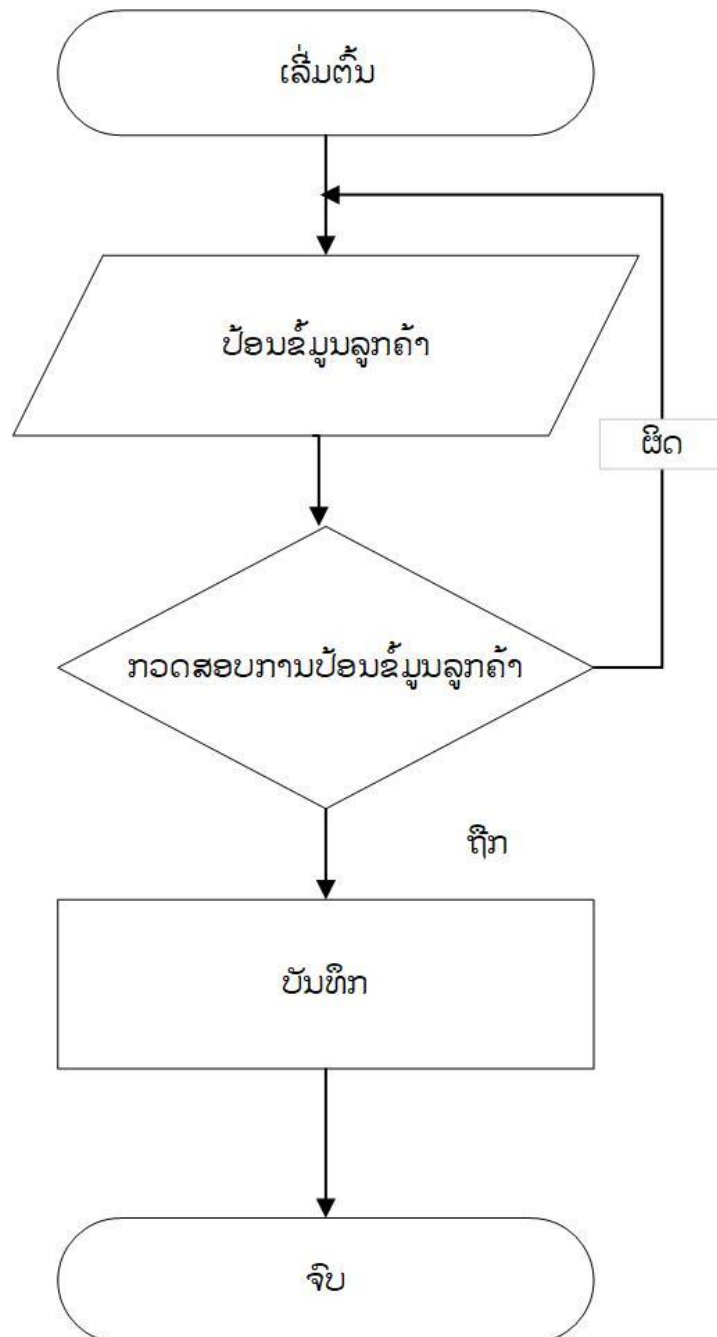
### 3.5.5 ແຜນວາດ Flowchart ການອອກປີ້



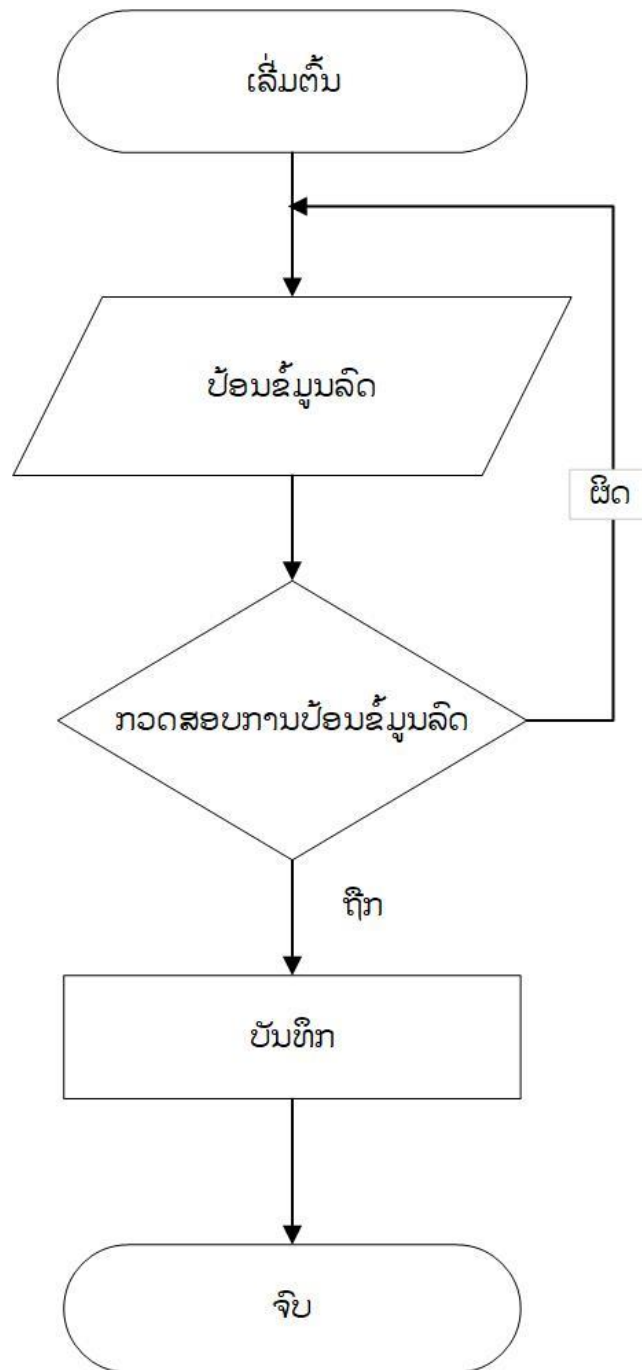
### 3.5.6 ແຜນວາດ Flowchart ຈັດການຂໍ້ມູນພະນັກງານ



### 3.5.7 ແຜນວາດ Flowchart ຈັດການຂໍ້ມູນລູກຄ້າ

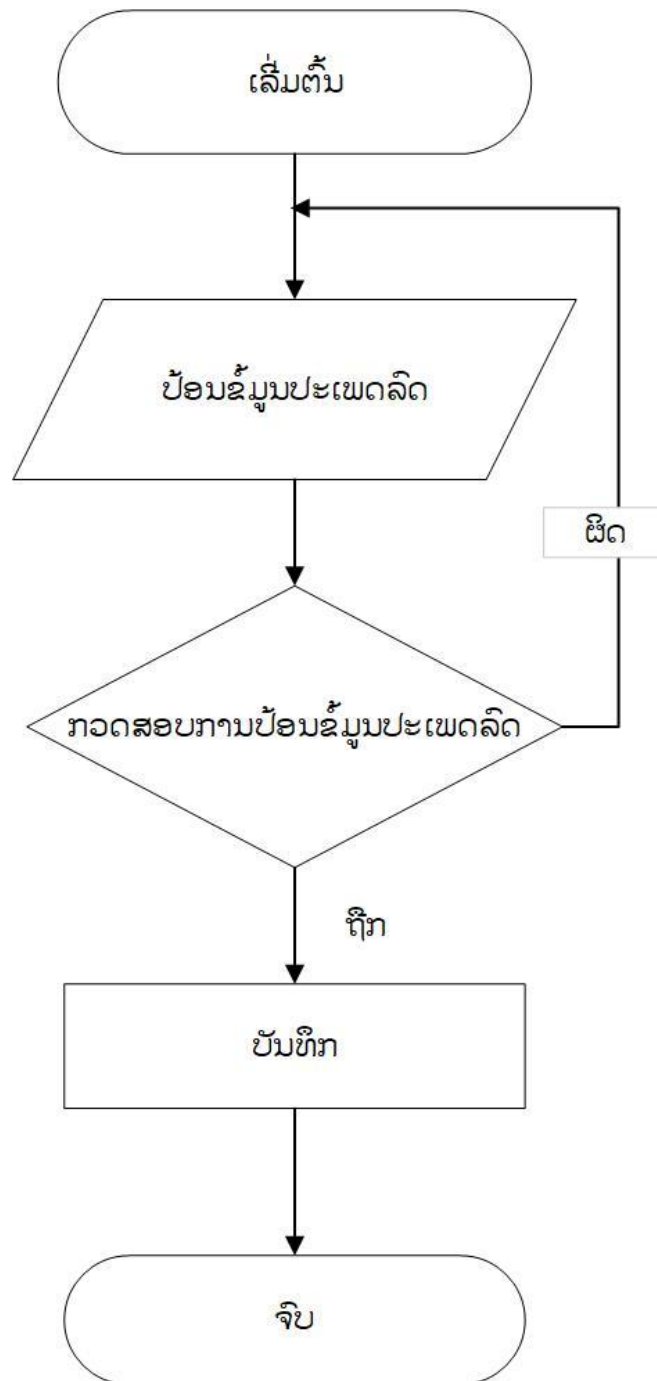


### 3.5.8 ແຜນວາດ Flowchart ຈັດການຂໍ້ມູນລົດ

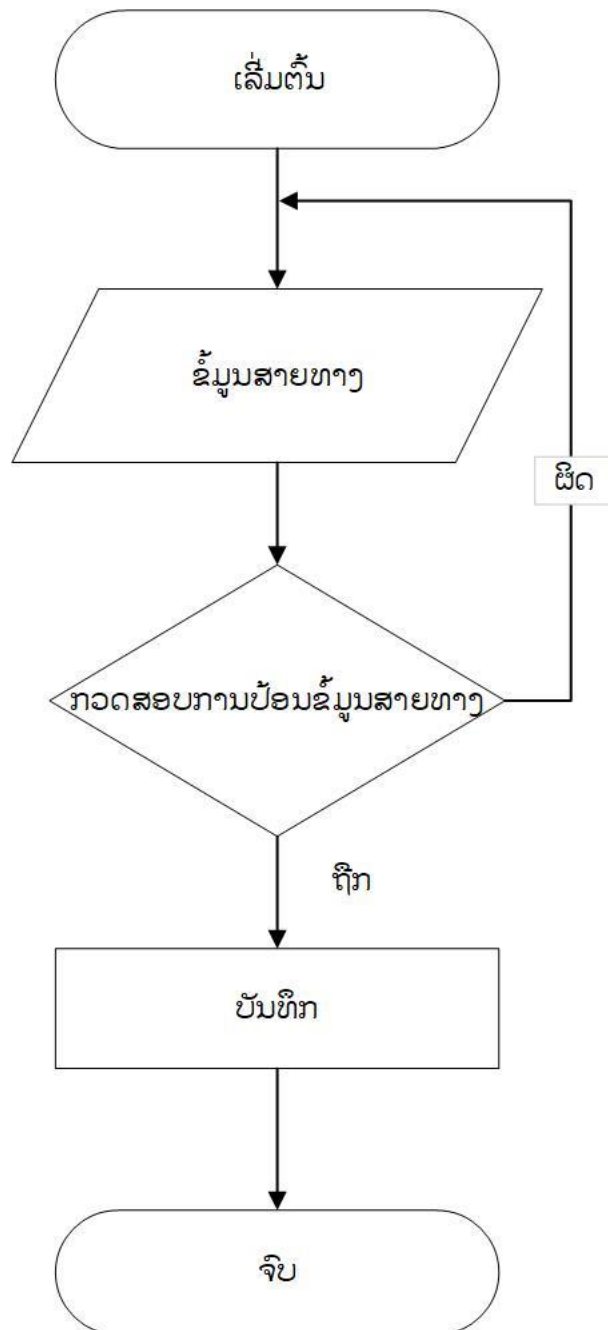




### 3.5.9 ແຜນວາດ Flowchart ຈັດການຂໍ້ມູນປະເພດລົດ



### 3.5.10 ແຜນວາດ Flowchart ຈັດການຂໍ້ມູນສາຍທາງ



## ບົດທີ 4

### ຜົນຂອງການວິເຄາະຂໍ້ມູນ ແລະ ການອະທິບາຍຜົນ

#### 4.1 ໜ້າຕ່າງການ (Login Form)

ເມື່ອເປີດໂປຣແກຣມຂຶ້ນມາກໍຈະເຫັນວ່າເຂົ້າສູ່ລະບົບເປັນໜ້າທຳອິດຈາກນັ້ນຈຶ່ງທຳການເຂົ້າສູ່ລະບົບດ້ວຍຊື່ ແລະ ລະຫັດເຂົ້າໃຊ້ເພື່ອເຂົ້າໄປດຳນິໃນຂອງໂປຣແກຣມດັ່ງຮູບລຸ່ມນີ້:

- ວິທີການເຂົ້າສູ່ລະບົບມີຄື :

- 1) ໃສ່ຊື່ຜູ້ໃຊ້ລະບົບ.
- 2) ໃສ່ລະຫັດຜ່ານ.
- 3) ກົດປຸ່ມເພື່ອເຂົ້າລະບົບ.
- 4) ຖ້າຊື່ກັບລະຫັດຖືກຈະເປີດນຳໃຊ້ໂປຣແກຣມໄດ້ປົກກະຕິແລ້ວ.
- 5) ຖ້າຊື່ກັບລະຫັດຜິດຈະມີຂໍ້ຄວາມຂຶ້ນມາວ່າ ກະລຸນາກວດສອບຂໍ້ມູນຂອງທ່ານໃຫ້ຖືກຕ້ອງ ແລ້ວກົດປຸ່ມ OK
- 6) ຖ້າບໍ່ເຂົ້າລະບົບແມ່ນກົດປຸ່ມອອກລະບົບ.

#### 4.2 ຟອມຫຼັກ (Main Form)

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

## ບົດທີ 5

### ສະຫຼຸບ ແລະ ຂໍ້ສະເໜີ

#### 5.1 ສະຫຼຸບ

ລະບົບຂາຍປີລິດເມອອນລາຍຂອງສະຖານນີຂົນສົ່ງໂດຍສານສາຍໃຕ້ແມ່ນຖືກພັດທະນາຂຶ້ນເພື່ອຊ່ວຍໃຫ້ສະຖານນີມີລະບົບທີ່ທັນສະໄໝ, ສະດວກສະບາຍ ແລະ ເຮັດໃຫ້ການຈັດການຂໍ້ມູນ, ການບໍລິການພາຍໃນສະຖານນີຢ່າງວ່ອງໄວ, ມີຄວາມເປັນລະບຽບ, ສະດວກໃນການລາຍງານ, ຂໍ້ມູນທີ່ມີຄວາມຖືກຕ້ອງ ແລະ ຊັດເຈນ. ຂອບເຂດຂອງການດຳເນີນວຽກງານໂປຣແກຣມ ຂອງພວກຂ້າພະເຈົ້າມີຈັດການຂໍ້ມູນ, ສະໝັກສະມາຊິກ, ບໍລິການ ແລະ ລາຍງານ. ເຊິ່ງເປັນການທົດແທນການເຮັດວຽກໃນລະບົບເກົ່າໂດຍລະບົບໃໝ່ເພື່ອຫຼຸດຜ່ອນບັນຫາການເສຍຫາຍຂອງຂໍ້ມູນ. ພວກນັ້ນສາມາດພັດທະນາໂປຣແກຣມເພື່ອຊ່ວຍໃຫ້ເຮັດວຽກສະດວກສະບາຍຍິ່ງຂຶ້ນ ແລະ ໄດ້ອອກແບບໜ້າຟອມການປ້ອນຂໍ້ມູນຕ່າງໆ.

- ຟອມຈັດການຂໍ້ມູນຫຼັກ 3 ຟອມ.
- ສ້າງຟອມຈັດການໄດ້ 5 ຟອມ.
- ຟອມການຄົ້ນຫາໄດ້ 5 ຟອມ.
- ພິມລາຍງານທັງໝົດໄດ້ 9 ລາຍງານ.

#### 5.2 ຈຸດດີ

- ຮູ້ວິເຄາະຫາບັນຫາ ແລະ ສາເຫດຂອງລະບົບເກົ່າ.
- ຈະໄດ້ລະບົບຈອງປີລິດເມສາຍໃຕ້ແບບອອນລາຍໃໝ່.
- ສາມາດນຳໃຊ້ເວບໄຊທີ່ສ້າງຂຶ້ນມາເຂົ້າຊ່ວຍໃນການຈອງປີລິດເມແບບອອນລາຍ.
- ເວບໄຊທີ່ສ້າງຂຶ້ນມາສາມາດຈອງອອນລາຍໄດ້.
- ການເຮັດບົດລາຍງານສະດວກສະບາຍ ແລະ ວ່ອງໄວຂຶ້ນ.

### 5.3 ຈຸດອ່ອນ

ໂປຣແກຣມນີ້ຍັງບໍ່ສົມບູນ ແລະ ຄົບຖ້ວນຕາມຄວາມຕ້ອງການຂອງສະຖານນີຂົນສົ່ງ ໂດຍສານສາຍໃຕ້ເທື່ອ ເນື່ອງຈາກວ່າໃນການສຶກສາຍັງບໍ່ທັນມີປະສົບການໃນການຂຽນເວບໄຊ ມາກ່ອນ, ຍັງບໍ່ຄວບຄຸມເຖິງຄວາມຕ້ອງການຕົວຈິງຂອງຜູ້ໃຊ້ລະບົບ.

### 5.4 ແນວທາງໃນການພັດທະນາ ແລະ ຂະຫຍາຍຕໍ່ຂອງສະຖານີ

ເນື່ອງຈາກວ່າໂປຣແກຣມນີ້ເປັນໂປຣແກຣມທີ່ສ້າງຂຶ້ນມາແລ້ວ ດັ່ງນັ້ນ ເພື່ອເປັນການກວດ ສວບຫາຂໍ້ພິດພາດ ແລະ ເພື່ອຫາຈຸດດີຈຸດອ່ອນມາທຳການປັບປຸງ, ແກ້ໄຂຈຶ່ງຄວນນຳເອົາໂປຣ ແກຣມນີ້ໄປປັບປຸງຈຸດບົກພ່ອງຂອງໂປຣແກຣມໃນບາງສ່ວນເພື່ອເຮັດໃຫ້ໂປຣແກຣມມີ ປະສິດທິພາບ ແລະ ເຮັດວຽກໄດ້ດີຂຶ້ນເພື່ອຈະນຳເອົາໄປປະກອບສ່ວນເຂົ້າໃນການເຮັດວຽກງານ ຕົວຈິງ.

ດັ່ງນັ້ນ ໂປຣແກຣມນີ້ເປັນໂປຣແກຣມໜຶ່ງທີ່ມີຄຸນສົມບັດໃນການໃຊ້ຖານຂໍ້ມູນໄດ້.

ເອກະສານອ້າງອີງ

## ປະຫວັດຫຍໍ້ຜູ້ຂຽນບົດ



ຊື່ ແລະ ນາມສະກຸນ: ທ້າວ ມະໂນພອນ ມະໂນກຸນ

ວັນ,ເດືອນ, ປີເກີດ: 16 ເດືອນ ທັນວາ ປີ 1995

ບ້ານເກີດ: ທົ່ງຂັນຄຳ ເມືອງ ຈັນທະບູລີ ແຂວງ ນະຄອນຫຼວງວຽງຈັນ

ບ້ານຢູ່ປັດຈຸບັນ: ທົ່ງຂັນຄຳ ເມືອງ ຈັນທະບູລີ ແຂວງ ນະຄອນຫຼວງວຽງຈັນ

ການສຶກສາ: ປີ 2016 ຈົບຊັ້ນສູງ ທີ່ ສະຖາບັນພັດທະນາສີມືແຮງງານລາວ - ເກົາຫຼີ

ປີ 2013 ຈົບມັດທະຍົມຕອນປາຍ: ທີ່ ມ.ສ ເຈົ້າອານຸວົງ

ປີ 2010 ຈົບມັດທະຍົມຕອນຕົ້ນ: ທີ່ ມ.ສ ເຈົ້າອານຸວົງ

ປີ 2007 ຈົບປະຖົມສົມບູນ: ທີ່ ໂຮງຮຽນປະຖົມສົມບູນອານຸ

ເບີໂທ: 020 5400 0003 , 020 2814 9278

ອີເມວ: [dou\\_2020@hotmail.com](mailto:dou_2020@hotmail.com)

ລາຍເຊັນເຈົ້າຂອງປະຫວັດ

ຊື່ແຈ້ງ.....

## ປະຫວັດຫຍໍ້ຜູ້ຂຽນບົດ



ຊື່ ແລະ ນາມສະກຸນ: ທ້າວ ພອນຄຳ ແກ້ວມະນີ

ວັນ,ເດືອນ,ປີເກີດ: 07 ເດືອນ ມີນາ ປີ 1995

ບ້ານເກີດ:ບ້ານ ທ້ວ, ເມືອງ: ວຽງພູຄາ, ແຂວງ: ຫຼວງນ້ຳທາ

ບ້ານຢູ່ປັດຈຸບັນ:ບ້ານ ທ້ວ, ເມືອງ: ວຽງພູຄາ, ແຂວງ: ຫຼວງນ້ຳທາ

ການສຶກສາ: ປີ 2016 ຈົບຊັ້ນສູງ: ທີ່ ສະຖາບັນພັດທະນາສີມືແຮງງານລາວ - ເກົາຫຼີ

ປີ 2013 ຈົບມັດທະຍົມສົມບູນ: ທີ່ ເມືອງວຽງພູຄາ

ປີ 2007 ຈົບປະຖົມສົມບູນ: ທີ່ ໂຮງຮຽນປະຖົມສົມບູນບ້ານດົງວຽງ

ເບີໂທ: 020 2802 2677

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

ລາຍເຊັນເຈົ້າຂອງປະຫວັດ

ຊື່ແຈ້ງ.....