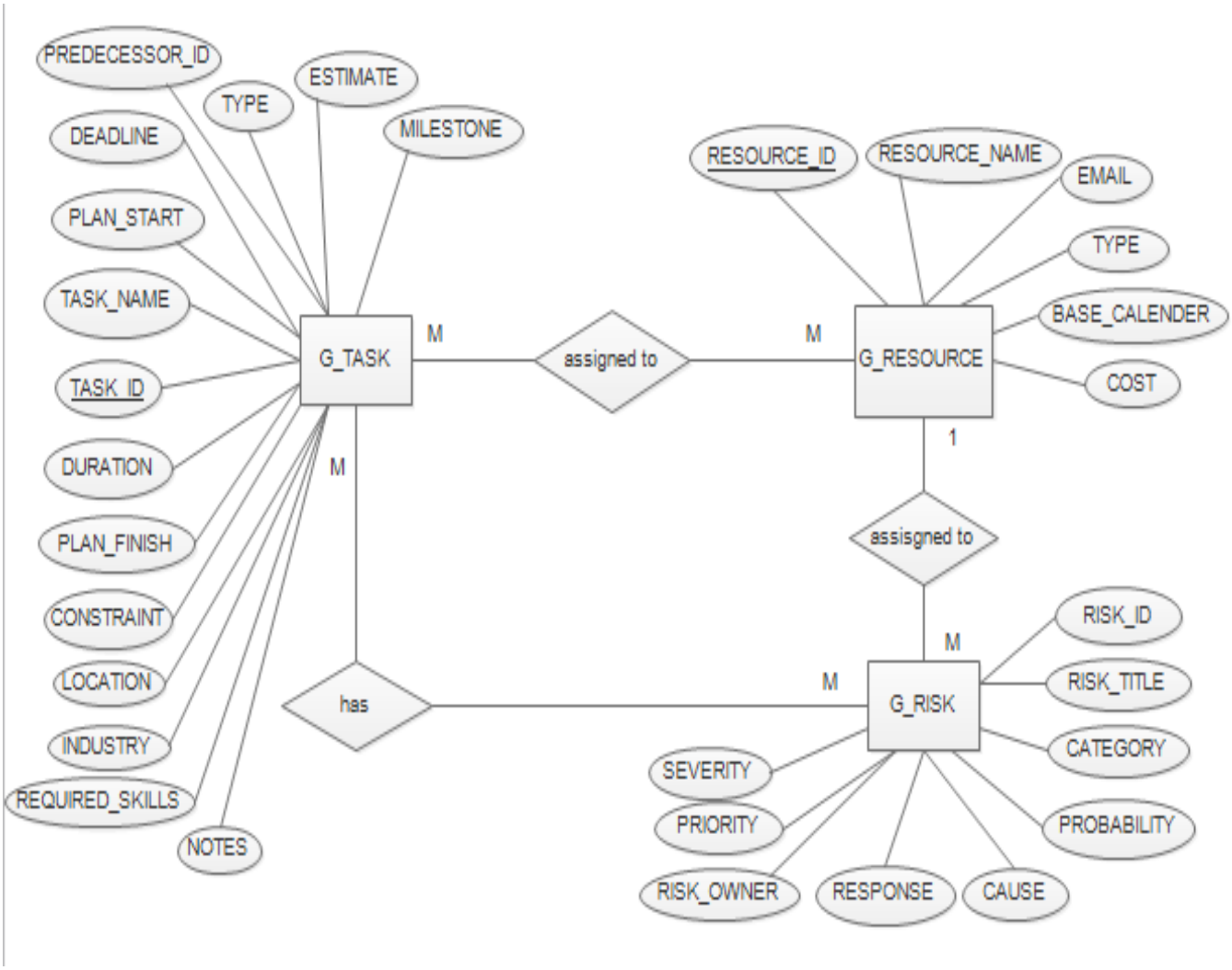
FA14: CMPE-281 Sec 02 – Cloud Technologies

Lab #2 – Multi-Tenant Domain Modeling

Group-180
Laxmi Phalak - 009394671
Minu Theresa Thomas - 009347221
Shivadeepthi Chilukuri - 009386351

Domain Analysis Diagrams:

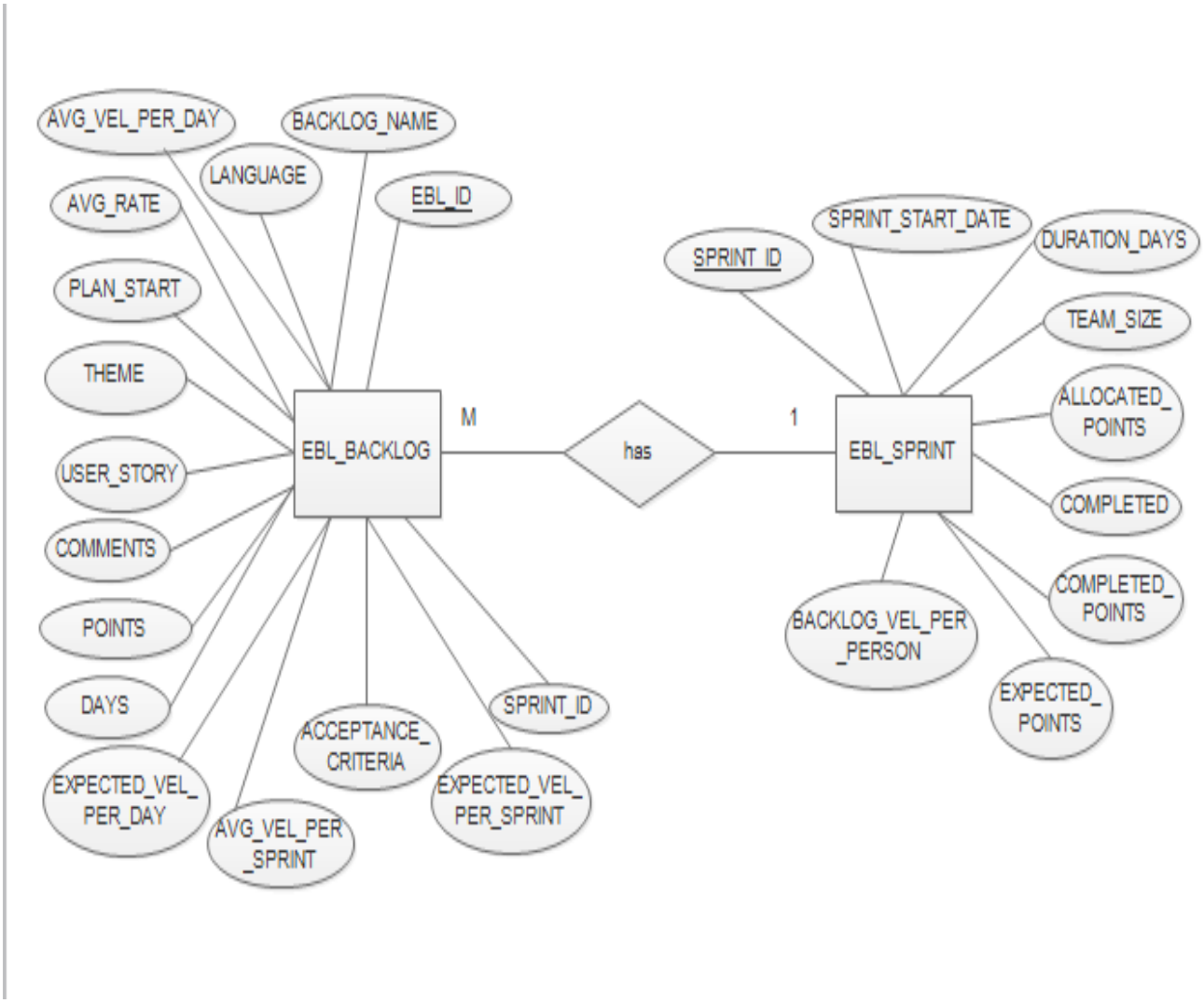
Entity Relationship Diagram for Gantter:



Annotations : Features observerd:

A Task can be assigned to many resources and a resource can take more than one task as well. Similar is the case with Risks and Tasks. So in Relational Data base, we are keeping two more table to normalize the above mentioned tables.

Entity Relationship Diagram for EasyBacklog:

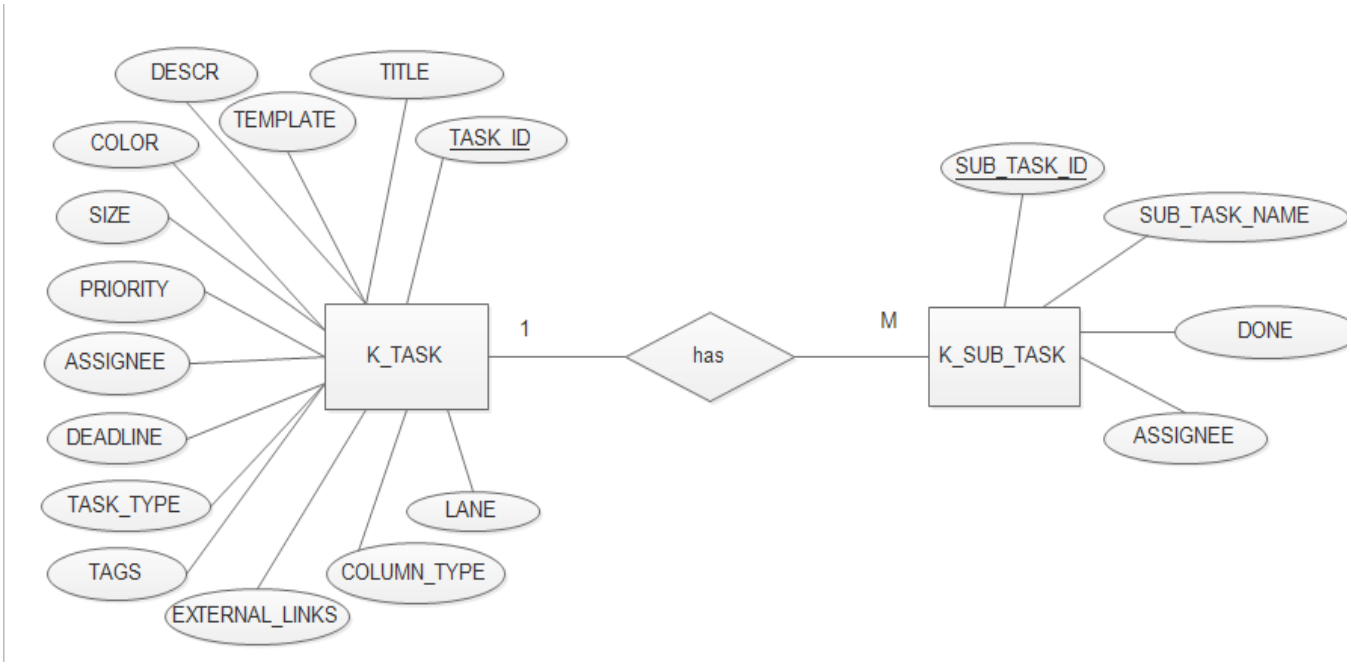


Features Observed :

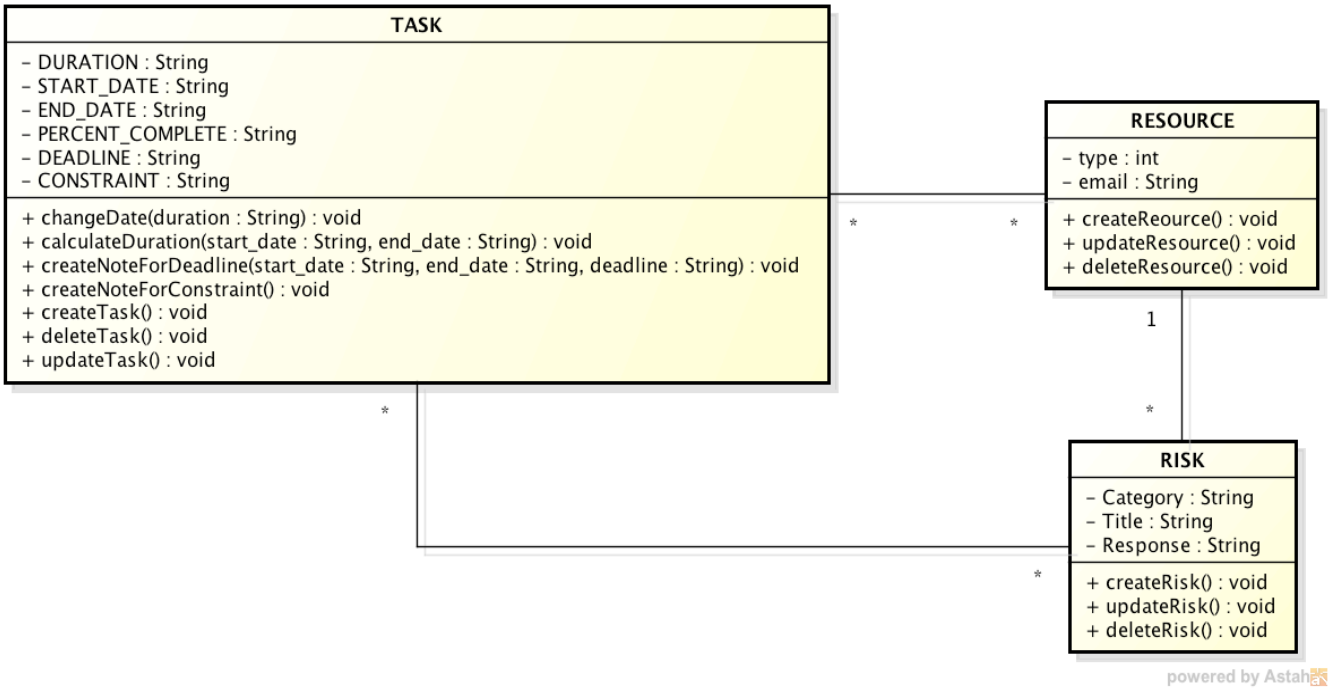
In one sprint multiple backlogs can be handled. Sprint has status which specifies whether a story has been completed or not.

In Kannanize : tasks can have more than one sub tasks.

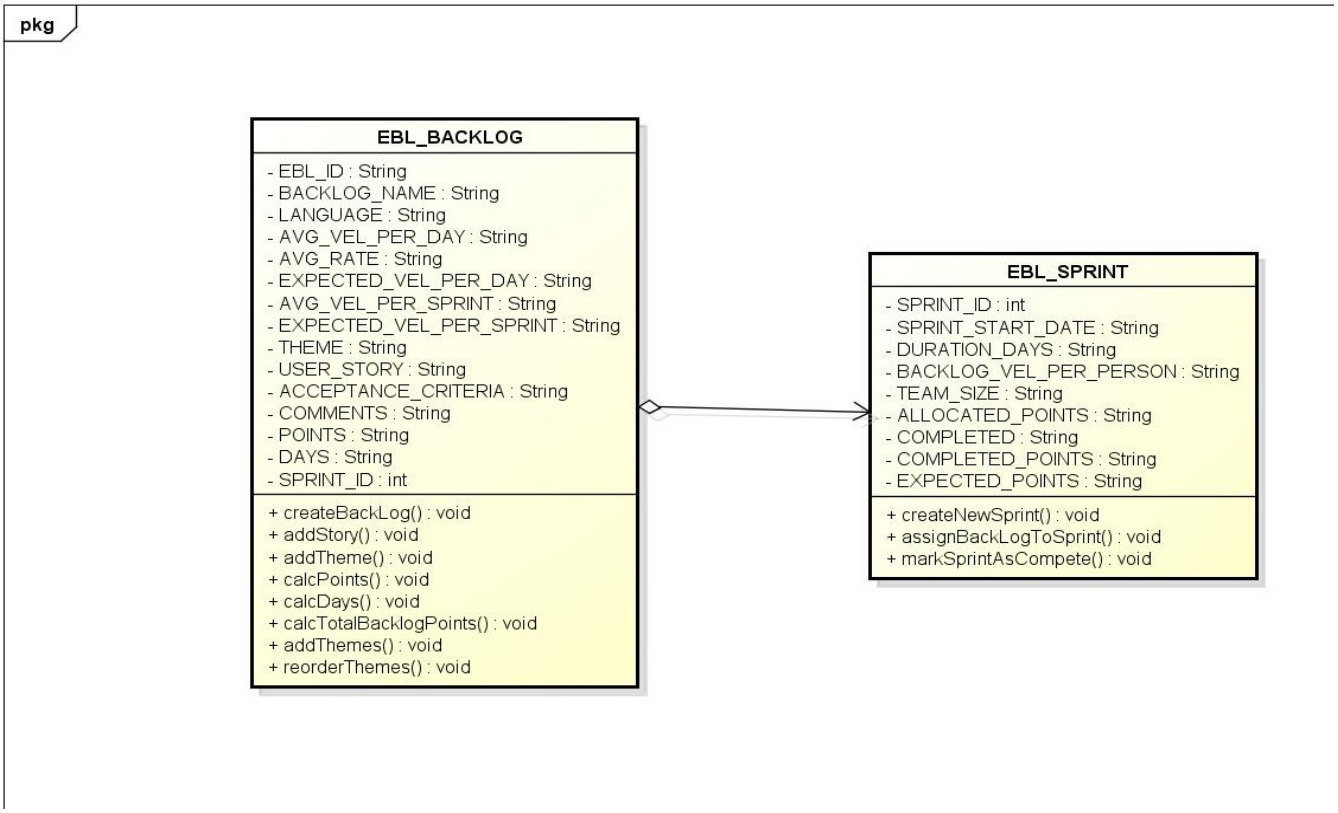
Entity Relationship Diagram for Kanbanize:



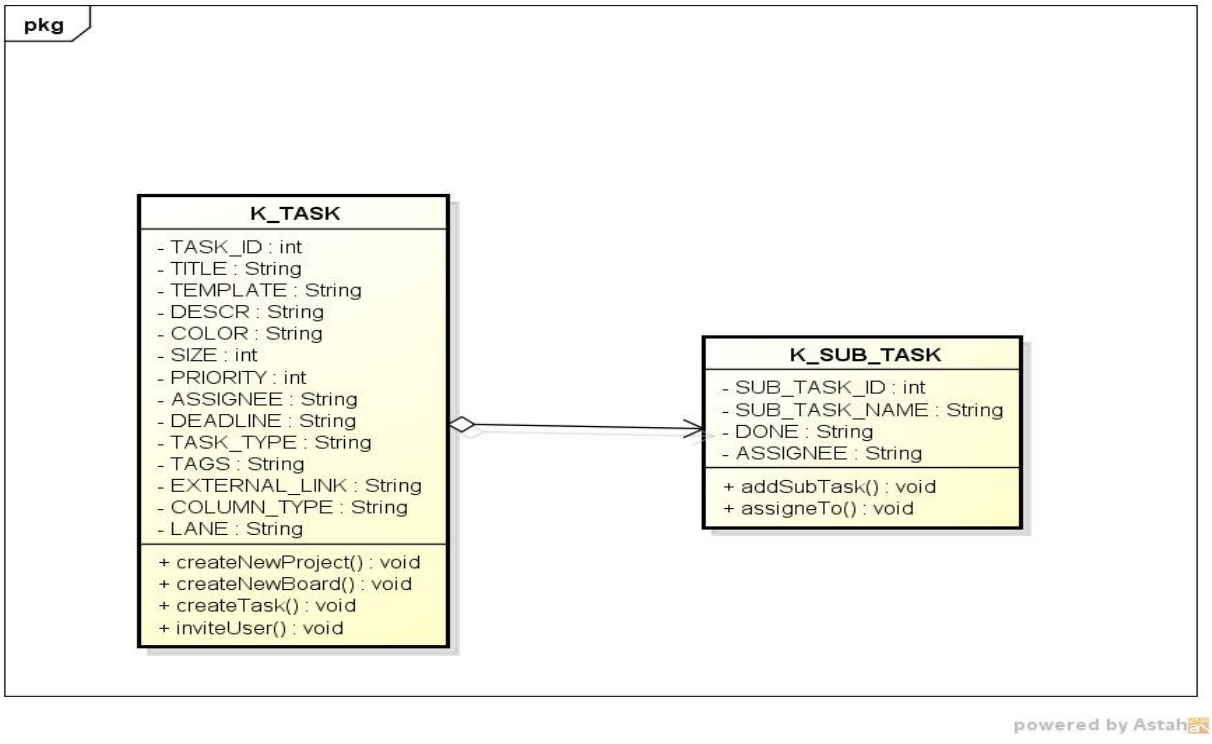
UML Diagram for Gantter:



UML Diagram for EasyBacklog:



UML Diagram for Kanbanize:



Feature Modeling:

Comparison of UI for all three SaaS Applications:

1. EasyBacklog:

In EasyBacklog, each task in the Sprint is represented as a Story. Each Story has a Backlog, and Sprints. There are tabs to show the progress. The tabs on the left side include Backlog, Stats (which shows the average velocity per day) and one tab each for the number of Sprints. In the Backlog tab, all stories are categorized into themes. Each theme is automatically assigned a theme code (shown below as 'HOP') which is used to prefix each story's ID within that theme.
In the Sprint tab, the stories for that particular Sprint are displayed along with the Sprint Points summary (Points allocated, Points expected and Points completed). The Points are also displayed in the right-hand side tabs.

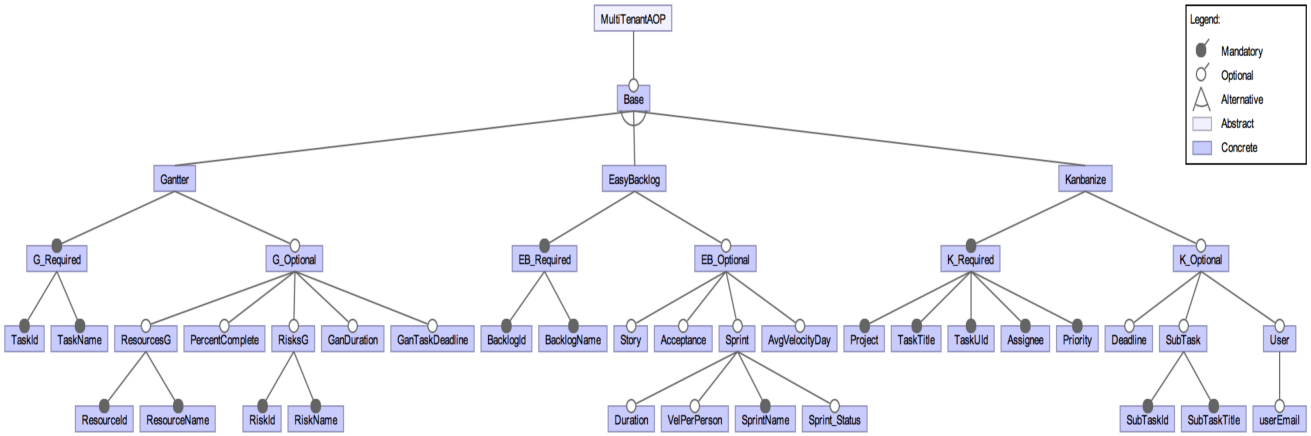
2. Gantter:

In Gantter, each task in the Sprint is represented as a Task. Each Task is assigned to a Resource and it has Risks. The format is like an Excel sheet where there are sheets for Tasks, Resources, Calendars and Risks. The Task sheet has columns for Duration, Start date, Finish date, Predecessors and Resources assigned to the Task. The Duration is also represented in a graphical timeline in the sheet. The Resources sheet has columns for Email, Type of Resource and Cost. The Risks sheet contains columns for Response, Cause of the Risk, Probability, Severity and Priority of the Risk.

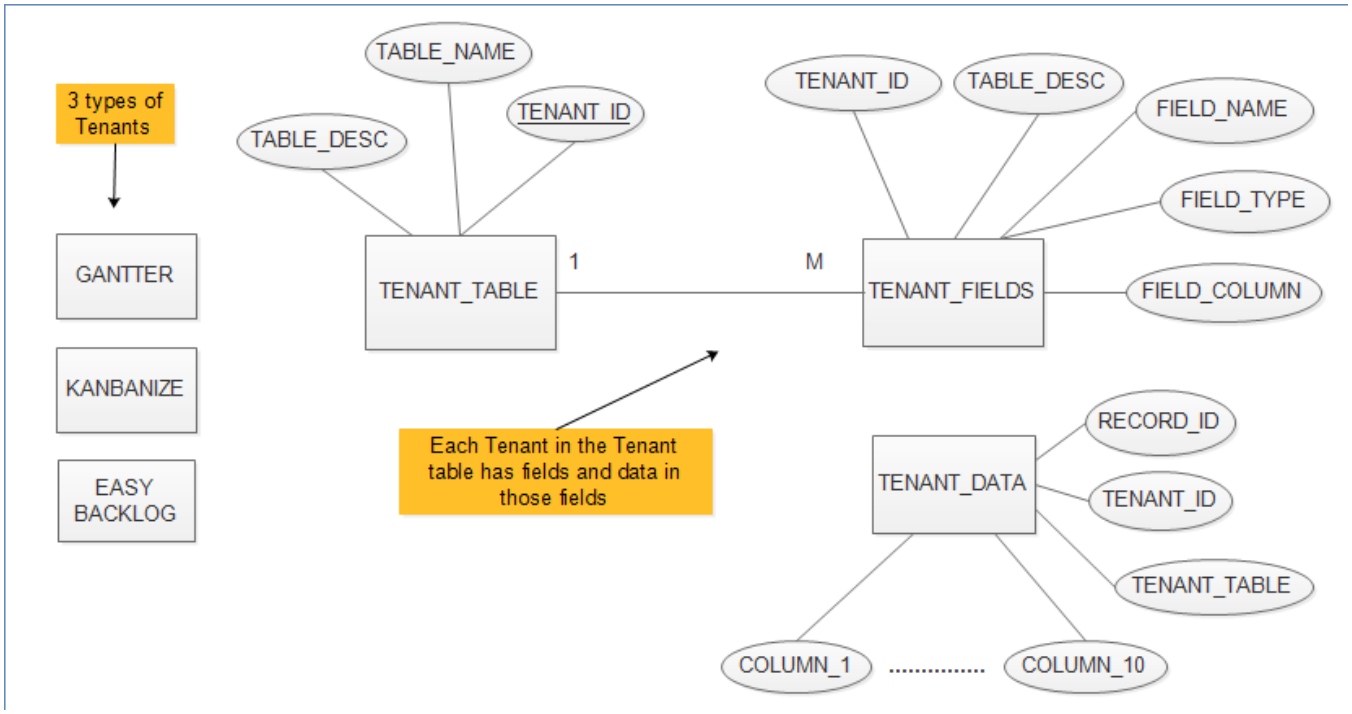
3. Kanbanize:

In Kanbanize, each task in the Sprint is represented as a Card. The format is like a Board where there are several Cards pinned up. The Card which represents a task can have many sub-tasks under it. Each Card has an Id and also the time for the task. You can define unlimited number of Kanban boards online. You can add columns, unlimited number of sub-columns as well as horizontal lanes to address priority, classes of work, customers, etc. You can also define roles for users and assign them for each board.

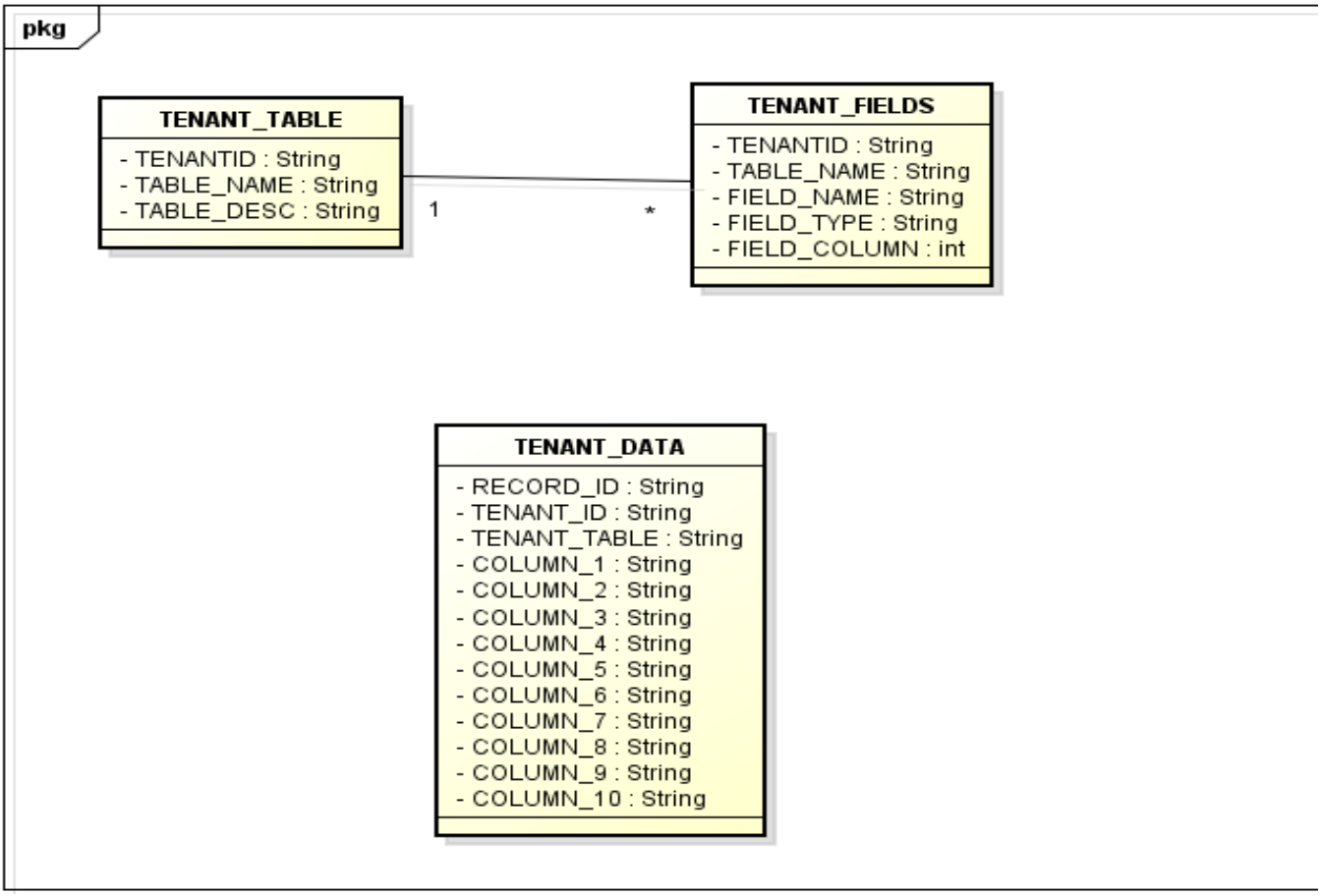
Feature Modeling



Final Multi-Tenant Entity Relationship Diagram:



Final Multi-Tenant UML Diagram:



MySQL Database Implementation:

Gantter:

```
CREATE TABLE IF NOT EXISTS `c9`.`G_TASK` (
  `TASK_ID` INT NOT NULL,
  `TASK_NAME` VARCHAR(45) NULL,
  `DURATION` VARCHAR(45) NULL,
  `PLAN_START` VARCHAR(45) NULL,
  `PLAN_FINISH` VARCHAR(45) NULL,
  `PREDECESSOR_ID` INT NULL,
  `RESOURCE_ID` INT NULL,
  `DEADLINE` VARCHAR(45) NULL,
  `CONSTRAINT` VARCHAR(45) NULL,
  `NOTES` VARCHAR(45) NULL,
  `INDUSTRY` VARCHAR(45) NULL,
  `REQUIRED_SKILLS` VARCHAR(45) NULL,
  `LOCATION` VARCHAR(45) NULL,
  `TYPE` VARCHAR(45) NULL,
  `ESTIMATE` VARCHAR(45) NULL,
  `MILESTONE` VARCHAR(45) NULL)
ENGINE = InnoDB;

CREATE TABLE IF NOT EXISTS `c9`.`G_TASK_RESOURCE` (
  `TASK_RESOURCE_ID` INT NOT NULL,
  `TASK_ID` INT NOT NULL,
  `RESOURCE_ID` INT NOT NULL)
ENGINE = InnoDB;

CREATE TABLE IF NOT EXISTS `c9`.`G_TASK_RISK` (
  `TASK_RISK_ID` INT NOT NULL,
  `TASK_ID` INT NOT NULL,
  `RISK_ID` INT NOT NULL)
ENGINE = InnoDB;

CREATE TABLE IF NOT EXISTS `c9`.`G_RESOURCE` (
  `RESOURCE_ID` INT NOT NULL,
  `RESOURCE_NAME` VARCHAR(45) NULL,
  `EMAIL` VARCHAR(45) NULL,
  `TYPE` VARCHAR(45) NULL,
  `COST` VARCHAR(45) NULL,
  `BASE_CALENDER` INT NULL)
ENGINE = InnoDB;

CREATE TABLE IF NOT EXISTS `c9`.`G_RISK` (
  `RISK_ID` INT NOT NULL,
  `RISK_TITLE` VARCHAR(45) NULL,
  `CATEGORY` VARCHAR(45) NULL,
  `RISK_OWNER` VARCHAR(45) NULL,
  `RESPONSE` VARCHAR(45) NULL,
  `CAUSE` VARCHAR(45) NULL,
  `PROBABILITY` VARCHAR(45) NULL,
  `SEVERITY` VARCHAR(45) NULL,
  `PRIORITY` VARCHAR(45) NULL)
ENGINE = InnoDB;
```

EasyBackLog:

```
CREATE TABLE IF NOT EXISTS `c9`.`EBL_BACKLOG` (
  `EBL_ID` INT NOT NULL,
  `BACKLOG_NAME` VARCHAR(45) NULL,
  `LANGUAGE` VARCHAR(45) NULL,
  `AVG_VEL_PER_DAY` VARCHAR(45) NULL,
  `AVG_RATE` VARCHAR(45) NULL,
  `EXPECTED_VEL_PER_DAY` VARCHAR(45) NULL,
  `AVG_VEL_PER_SPRINT` VARCHAR(45) NULL,
  `EXPECTED_VEL_PER_SPRINT` VARCHAR(45) NULL,
  `THEME` VARCHAR(45) NULL,
  `USER_STORY` VARCHAR(45) NULL,
  `ACCEPTANCE_CRITERIA` VARCHAR(45) NULL,
  `COMMENTS` VARCHAR(45) NULL,
  `POINTS` VARCHAR(45) NULL,
  `DAYS` VARCHAR(45) NULL,
  `SPRINT_ID` INT NULL)
ENGINE = InnoDB;


CREATE TABLE IF NOT EXISTS `c9`.`EBL_SPRINT` (
  `SPRINT_ID` INT NOT NULL,
  `SPRINT_START_DATE` VARCHAR(45) NULL,
  `DURATION_DAYS` VARCHAR(45) NULL,
  `BACKLOG_VEL_PER_PERSON` VARCHAR(45) NULL,
  `TEAM_SIZE` VARCHAR(45) NULL,
  `ALLOCATED_POINTS` VARCHAR(45) NULL,
  `COMPLETED` VARCHAR(45) NULL,
  `COMPLETED_POINTS` VARCHAR(45) NULL,
  `EXPECTED_POINTS` VARCHAR(45) NULL)
ENGINE = InnoDB;
```

Kanbanize:

```
CREATE TABLE IF NOT EXISTS `c9`.`K_TASK` (
  `TASK_ID` INT NOT NULL,
  `TITLE` VARCHAR(45) NULL,
  `TEMPLATE` VARCHAR(45) NULL,
  `DESCR` VARCHAR(45) NULL,
  `COLOR` VARCHAR(45) NULL,
  `SIZE` INT NULL,
  `PRIORITY` INT NULL,
  `ASSIGNEE` VARCHAR(45) NULL,
  `DEADLINE` VARCHAR(45) NULL,
  `TASK_TYPE` VARCHAR(45) NULL,
  `TAGS` VARCHAR(45) NULL,
  `EXTERNAL_LINK` VARCHAR(45) NULL,
  `COLUMN_TYPE` VARCHAR(45) NULL,
  `LANE` VARCHAR(45) NULL
  )
ENGINE = InnoDB;

CREATE TABLE IF NOT EXISTS `c9`.`K_SUB_TASK` (
  `SUB_TASK_ID` INT NOT NULL,
```

```
 `SUB_TASK_NAME` VARCHAR(45) NULL,
 `DONE` VARCHAR(45) NULL,
 `ASSIGNEE` VARCHAR(45) NULL)
ENGINE = InnoDB;
```

Tenant Table(SaaS) : DDL to create a single Multi-Tenant Schema in MySQL

```
CREATE TABLE IF NOT EXISTS `c9`.`TENANT_TABLE` (
 `TENANT_ID` VARCHAR(10) NOT NULL,
 `TABLE_NAME` VARCHAR(45) NOT NULL,
 `TABLE_DESC` VARCHAR(80) NULL,
 PRIMARY KEY (`TENANT_ID`, `TABLE_NAME`))
ENGINE = InnoDB;
```

Tenant Fields

```
CREATE TABLE IF NOT EXISTS `c9`.`TENANT_FIELDS` (
 `TENANT_ID` VARCHAR(10) NOT NULL,
 `TABLE_NAME` VARCHAR(45) NOT NULL,
 `FIELD_NAME` VARCHAR(45) NOT NULL,
 `FIELD_TYPE` VARCHAR(80) NULL,
 `FIELD_COLUMN` INT NOT NULL,
 PRIMARY KEY (`TENANT_ID`, `TABLE_NAME`, `FIELD_NAME`),
 CONSTRAINT `fk_TENANT_FIELDS_TENANT_TABLE`
  FOREIGN KEY (`TENANT_ID` , `TABLE_NAME`)
  REFERENCES `c9`.`TENANT_TABLE` (`TENANT_ID` , `TABLE_NAME`)
  ON DELETE NO ACTION
  ON UPDATE NO ACTION)
ENGINE = InnoDB;
```

Tenant Data

```
CREATE TABLE IF NOT EXISTS `c9`.`TENANT_DATA` (
 `RECORD_ID` VARCHAR(45) NOT NULL,
 `TENANT_ID` VARCHAR(10) NOT NULL,
 `TENANT_TABLE` VARCHAR(45) NULL,
 `COLUMN_1` VARCHAR(80) NULL,
 `COLUMN_2` VARCHAR(80) NULL,
 `COLUMN_3` VARCHAR(80) NULL,
 `COLUMN_4` VARCHAR(80) NULL,
 `COLUMN_5` VARCHAR(80) NULL,
 `COLUMN_6` VARCHAR(80) NULL,
 `COLUMN_7` VARCHAR(80) NULL,
 `COLUMN_8` VARCHAR(80) NULL,
 `COLUMN_9` VARCHAR(80) NULL,
 `COLUMN_10` VARCHAR(80) NULL,
 `COLUMN_11` VARCHAR(80) NULL,
 `COLUMN_12` VARCHAR(80) NULL,
 `COLUMN_13` VARCHAR(80) NULL,
 `COLUMN_14` VARCHAR(80) NULL,
 `COLUMN_15` VARCHAR(80) NULL,
 `COLUMN_16` VARCHAR(80) NULL,
 PRIMARY KEY (`TENANT_ID`, `RECORD_ID`))
ENGINE = InnoDB;
```

DML (Insert Statements) to provision the Tenant-Specific Schemas.

1. Populate TENANT_TABLE for three different SaaS applications:

```
insert into TENANT_TABLE( TENANT_ID, TABLE_NAME, TABLE_DESC )
values ( 'GA', 'G_TASK', 'Task Table for Gantter' ) ;
insert into TENANT_TABLE( TENANT_ID, TABLE_NAME, TABLE_DESC )
values ( 'GA', 'G_RESOURCE', 'Resource Table for Gantter' ) ;
insert into TENANT_TABLE( TENANT_ID, TABLE_NAME, TABLE_DESC )
values ( 'GA', 'G_RISK', 'Risk Table for Gantter' ) ;
insert into TENANT_TABLE( TENANT_ID, TABLE_NAME, TABLE_DESC )
values ( 'EBL', 'EBL_BACKLOG', 'Backlog Table for Backlog' ) ;
insert into TENANT_TABLE( TENANT_ID, TABLE_NAME, TABLE_DESC )
values ( 'EBL', 'EBL_SPRINT', 'Sprint Table for Backlog' ) ;
insert into TENANT_TABLE( TENANT_ID, TABLE_NAME, TABLE_DESC )
values ( 'KA', 'K_TASK', 'Task Table for Kanbanize' ) ;
insert into TENANT_TABLE( TENANT_ID, TABLE_NAME, TABLE_DESC )
values ( 'KA', 'K_SUB_TASK', 'Sub-Task Table for Kanbanize' ) ;

update TENANT_TABLE set TABLE_DESC = 'Task Table for Kanbanize' where tenant_id = 'KA' and
table_name = 'K_TASK';
update TENANT_TABLE set TABLE_DESC = 'Sub-Task Table for Kanbanize' where tenant_id = 'KA'
and table_name = 'K_SUB_TASK';
```

2. Populate TENANT_FIELDS for each application in SaaS multi-tenant model:

```
/*G_TASK*/
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ('GA', 'G_TASK', 'TASK_ID', 'INT', 1 ) ;
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'GA', 'G_TASK', 'TASK_NAME', 'VARCHAR(45)', 2 ) ;
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'GA', 'G_TASK', 'DURATION', 'VARCHAR(45)', 3 ) ;
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'GA', 'G_TASK', 'PLAN_START', 'VARCHAR(45)', 4 ) ;
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'GA', 'G_TASK', 'PLAN_FINISH', 'VARCHAR(45)', 5 ) ;
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'GA', 'G_TASK', 'PREDECESSOR_ID', 'INT', 6 ) ;
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'GA', 'G_TASK', 'RESOURCE_ID', 'INT', 7 ) ;
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'GA', 'G_TASK', 'DEADLINE', 'VARCHAR(45)',8 ) ;
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'GA', 'G_TASK', 'CONSTRAINT', 'VARCHAR(45)', 9 ) ;
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'GA', 'G_TASK', 'NOTES', 'VARCHAR(45)', 10 ) ;
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'GA', 'G_TASK', 'INDUSTRY', 'VARCHAR(45)', 11 ) ;
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'GA', 'G_TASK', 'REQUIRED_SKILLS', 'VARCHAR(45)', 12 ) ;
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'GA', 'G_TASK', 'LOCATION', 'VARCHAR(45)', 13 ) ;
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'GA', 'G_TASK', 'TYPE', 'VARCHAR(45)', 14 ) ;
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'GA', 'G_TASK', 'ESTIMATE', 'VARCHAR(45)', 15 ) ;
```

```
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'GA', 'G_TASK', 'MILESTONE', 'VARCHAR(45)',16 ) ;

/*G_RESOURCE*/
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'GA', 'G_RESOURCE', 'RESOURCE_ID', 'INT', 1 ) ;
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'GA', 'G_RESOURCE', 'RESOURCE_NAME', 'VARCHAR(45)', 2 ) ;
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'GA', 'G_RESOURCE', 'EMAIL', 'VARCHAR(45)', 3 ) ;
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'GA', 'G_RESOURCE', 'TYPE', 'VARCHAR(45)', 4 ) ;
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'GA', 'G_RESOURCE', 'COST', 'VARCHAR(45)', 5 ) ;
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'GA', 'G_RESOURCE', 'BASE_CALENDER', 'INT', 6 ) ;

/*G_RISK*/
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'GA', 'G_RISK', 'RISK_ID', 'INT', 1 ) ;
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'GA', 'G_RISK', 'RISK_TITLE', 'VARCHAR(45)', 2 ) ;
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'GA', 'G_RISK', 'CATEGORY', 'VARCHAR(45)', 3 ) ;
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'GA', 'G_RISK','RISK_OWNER', 'VARCHAR(45)', 4 ) ;
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'GA', 'G_RISK', 'RESPONSE', 'VARCHAR(45)', 5 ) ;
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'GA', 'G_RISK', 'CAUSE', 'VARCHAR(45)', 6 ) ;
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'GA', 'G_RISK', 'PROBABILITY', 'VARCHAR(45)', 7 ) ;
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'GA', 'G_RISK', 'SEVERITY', 'VARCHAR(45)',8 ) ;
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'GA', 'G_RISK', 'PRIORITY', 'VARCHAR(45)', 9 ) ;

/*K_TASK*/
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'KA', 'K_TASK', 'TASK_ID', 'INT', 1 ) ;
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'KA', 'K_TASK', 'TITLE', 'VARCHAR(45)', 2 ) ;
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'KA', 'K_TASK', 'TEMPLATE', 'VARCHAR(45)', 3 ) ;
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'KA', 'K_TASK', 'DESCR', 'VARCHAR(45)', 4 ) ;
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'KA', 'K_TASK', 'COLOR', 'VARCHAR(45)', 5 ) ;
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'KA', 'K_TASK', 'SIZE', 'INT', 6 ) ;
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'KA', 'K_TASK', 'PRIORITY', 'INT', 7 ) ;
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'KA', 'K_TASK', 'ASSIGNEE', 'VARCHAR(45)',8 ) ;
```

```
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'KA', 'K_TASK', 'DEADLINE', 'VARCHAR(45)', 9 ) ;
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'KA', 'K_TASK', 'TASK_TYPE', 'VARCHAR(45)', 10 ) ;
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'KA', 'K_TASK', 'TAGS', 'VARCHAR(45)', 11 ) ;
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'KA', 'K_TASK', 'EXTERNAL_LINK', 'VARCHAR(45)', 12 ) ;
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'KA', 'K_TASK', 'COLUMN_TYPE', 'VARCHAR(45)', 13 ) ;
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'KA', 'K_TASK', 'LANE', 'VARCHAR(45)', 14 ) ;

/*K_SUB_TASK*/
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'KA', 'K_SUB_TASK', 'SUB_TASK_ID', 'INT', 1 ) ;
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'KA', 'K_SUB_TASK', 'SUB_TASK_NAME', 'VARCHAR(45)', 2 ) ;
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'KA', 'K_SUB_TASK', 'DONE', 'VARCHAR(45)', 3 ) ;
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'KA', 'K_SUB_TASK', 'ASSIGNEE', 'VARCHAR(45)', 4 ) ;

/* EBL_BACKLOG*/
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'EBL', 'EBL_BACKLOG', 'EBL_ID', 'INT', 1 ) ;
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'EBL', 'EBL_BACKLOG', 'BACKLOG_NAME', 'VARCHAR(45)', 2 ) ;
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'EBL', 'EBL_BACKLOG', 'LANGUAGE', 'VARCHAR(45)', 3 ) ;
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'EBL', 'EBL_BACKLOG', 'AVG_VEL_PER_DAY', 'VARCHAR(45)', 4 ) ;
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'EBL', 'EBL_BACKLOG', 'AVG_RATE', 'VARCHAR(45)', 5 ) ;
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'EBL', 'EBL_BACKLOG', 'EXPECTED_VEL_PER_DAY', 'VARCHAR(45)', 6 ) ;
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'EBL', 'EBL_BACKLOG', 'AVG_VEL_PER_SPRINT', 'VARCHAR(45)', 7 ) ;
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'EBL', 'EBL_BACKLOG', 'EXPECTED_VEL_PER_SPRINT', 'VARCHAR(45)', 8 )
;
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'EBL', 'EBL_BACKLOG', 'THEME', 'VARCHAR(45)', 9 ) ;
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'EBL', 'EBL_BACKLOG', 'USER_STORY', 'VARCHAR(45)', 10 ) ;
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'EBL', 'EBL_BACKLOG', 'ACCEPTANCE_CRITERIA', 'VARCHAR(45)', 11 ) ;
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'EBL', 'EBL_BACKLOG', 'COMMENTS', 'VARCHAR(45)', 12 ) ;
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'EBL', 'EBL_BACKLOG', 'POINTS', 'VARCHAR(45)', 13 ) ;
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'EBL', 'EBL_BACKLOG', 'DAYS', 'VARCHAR(45)', 14 ) ;
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'EBL', 'EBL_BACKLOG', 'SPRINT_ID', 'INT', 15 ) ;
```

```
/* EBL_SPRINT*/
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'EBL','EBL_SPRINT', 'SPRINT_ID', 'INT', 1 ) ;
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'EBL','EBL_SPRINT', 'SPRINT_START_DATE', 'VARCHAR(45)', 2 ) ;
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'EBL','EBL_SPRINT', 'DURATION_DAYS', 'VARCHAR(45)', 3 ) ;
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'EBL','EBL_SPRINT', 'BACKLOG_VEL_PER_PERSON', 'VARCHAR(45)', 4 ) ;
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'EBL','EBL_SPRINT', 'TEAM_SIZE', 'VARCHAR(45)', 5 ) ;
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'EBL','EBL_SPRINT', 'ALLOCATED_POINTS', 'VARCHAR(45)', 6 ) ;
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'EBL','EBL_SPRINT', 'COMPLETED', 'VARCHAR(45)', 7 ) ;
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'EBL','EBL_SPRINT', 'COMPLETED_POINTS', 'VARCHAR(45)', 8 ) ;
insert into TENANT_FIELDS( TENANT_ID, TABLE_NAME, FIELD_NAME, FIELD_TYPE,
FIELD_COLUMN ) values ( 'EBL','EBL_SPRINT', 'EXPECTED_POINTS', 'VARCHAR(45)', 9 ) ;

insert into TENANT_DATA (RECORD_ID, TENANT_ID, TENANT_TABLE, COLUMN_1, COLUMN_2,
COLUMN_3, COLUMN_4, COLUMN_5, COLUMN_6  )
values ( '1001', 'EBL', 'EBL_BACKLOG', '1001', 'Design Data Models', 'Team283', '2014-09-01', '2014-
10-01', 'FALSE' ) ;
```

SQL queries that generates a listing of the tenant specific data elements and type.

```
-- QUERY
select T.TABLE_NAME, F.FIELD_NAME, F.FIELD_TYPE, F.FIELD_COLUMN
from TENANT_TABLE T, TENANT_FIELDS F
where T.TENANT_ID = F.TENANT_ID
and T.TENANT_ID = 'GA'
order by F.FIELD_COLUMN
```

AOP / Feature IDE Implementation:

MultiTenantSaas.java

```
public class MultiTenantSaas {

	public static void main(String args[])
	{
	   GantterTask gt = new GantterTask();
	   KanbanTask kt = new KanbanTask();
	   EasyBacklog eb = new EasyBacklog();

	   System.out.println("Welcome to multitenant world!");
	   createTask(gt);
	   updateTask(gt);
	   System.out.println("The updated Gantter task is : ");
	   System.out.println(gt.getTaskName());
	   System.out.println(gt.getPercentComplete());
	   System.out.println(gt.getGDuration());
	   createBackLog(eb);
	   updateBackLog(eb);
	   System.out.println("The updated  Easy Backlog is : ");
	   System.out.println(eb.getBackLogName());
	   System.out.println(eb.getStory());
	   System.out.println(eb.getAcceptance());
	   createKanbanTask(kt);
	   updateKanbanTask(kt);
	   System.out.println("The updated Kanban task is : ");
	   System.out.println(kt.getKanTaskName());
	   System.out.println(kt.getKanPriority());
	   System.out.println(kt.getAssignee());
	}

	private static void updateKanbanTask(KanbanTask kt) {
	   // TODO Auto-generated method stub

	}

	private static void updateBackLog(EasyBacklog eb) {
	   // TODO Auto-generated method stub

	}

	private static void updateTask(GantterTask gt) {
	   // TODO Auto-generated method stub

	}

	private static void createKanbanTask(KanbanTask kt) {
	   // TODO Auto-generated method stub

	}

	private static void createBackLog(EasyBacklog eb) {
```

```
        // TODO Auto-generated method stub

    }

    private static void createTask(GantterTask gt) {
        // TODO Auto-generated method stub

    }
}
```

GantterTask.java

```
import java.util.Date;
public class GantterTask {
    private int taskId;
    private String taskName;
    private float percentComplete;
    private int gDuration;
    private Date gDeadLine;
    private Resources resource;
    private Risks risk;
    private Date startdate;
    private Date endDate;

    public Date getStartdate() {
        return startdate;
    }
    public void setStartdate(Date startdate) {
        this.startdate = startdate;
    }
    public Date getEndDate() {
        return endDate;
    }
    public void setEndDate(Date endDate) {
        this.endDate = endDate;
    }
    public int getTaskId() {
        return taskId;
    }
    public void setTaskId(int taskId) {
        this.taskId = taskId;
    }
    public String getTaskName() {
        return taskName;
    }
    public void setTaskName(String taskName) {
        this.taskName = taskName;
    }
    public float getPercentComplete() {
        return percentComplete;
    }
    public void setPercentComplete(float percentComplete) {
        this.percentComplete = percentComplete;
    }
    public int getgDuration() {
```

```
        return gDuration;
    }
    public void setgDuration(int gDuration) {
        this.gDuration = gDuration;
    }
    public Date getgDeadLine() {
        return gDeadLine;
    }
    public void setgDeadLine(Date gDeadLine) {
        this.gDeadLine = gDeadLine;
    }
    public Resources getResource() {
        return resource;
    }
    public void setResource(Resources resource) {
        this.resource = resource;
    }
    public Risks getRisk() {
        return risk;
    }
    public void setRisk(Risks risk) {
        this.risk = risk;
    }

}
```

Resources.java

```
public class Resources {
    private int ResourceId;
    private String ResourceName;

    public int getResourceId() {
        return ResourceId;
    }
    public void setResourceId(int resourceId) {
        ResourceId = resourceId;
    }
    public String getResourceName() {
        return ResourceName;
    }
    public void setResourceName(String resourceName) {
        ResourceName = resourceName;
    }

}
```

Risks.java

```
public class Risks {
    private int riskId;
    private String riskName;
    public int getRiskId() {
        return riskId;
    }
```

```
    public void setRiskId(int riskId) {
        this.riskId = riskId;
    }
    public String getRiskName() {
        return riskName;
    }
    public void setRiskName(String riskName) {
        this.riskName = riskName;
    }

}
```

EasyBacklog.java

```
public class EasyBacklog {
    private int backlogId;
    private String backlogName;
    private String story;
    private String acceptance;
    private float avgVelPerDay;
    private Sprint sprint;
    public int getBacklogId() {
        return backlogId;
    }
    public void setBacklogId(int backlogId) {
        this.backlogId = backlogId;
    }
    public String getBacklogName() {
        return backlogName;
    }
    public void setBacklogName(String backlogName) {
        this.backlogName = backlogName;
    }
    public String getStory() {
        return story;
    }
    public void setStory(String story) {
        this.story = story;
    }
    public String getAcceptance() {
        return acceptance;
    }
    public void setAcceptance(String acceptance) {
        this.acceptance = acceptance;
    }
    public float getAvgVelPerDay() {
        return avgVelPerDay;
    }
    public void setAvgVelPerDay(float avgVelPerDay) {
        this.avgVelPerDay = avgVelPerDay;
    }
    public Sprint getSprint() {
        return sprint;
    }
    public void setSprint(Sprint sprint) {
```

```
        this.sprint = sprint;
    }

}
```

Sprint.java

```
public class Sprint {

    private int duration;
    private float velPerPerson;
    private String sprintName;
    public int getDuration() {
        return duration;
    }
    public void setDuration(int duration) {
        this.duration = duration;
    }
    public float getVelPerPerson() {
        return velPerPerson;
    }
    public void setVelPerPerson(float velPerPerson) {
        this.velPerPerson = velPerPerson;
    }
    public String getSprintName() {
        return sprintName;
    }
    public void setSprintName(String sprintName) {
        this.sprintName = sprintName;
    }

}
```

KanbanTask.java

```
import java.util.Date;
public class KanbanTask {
    private String projectName;
    private String kanTaskName;
    private int kanTaskId;
    private String assignee;
    private String kanPriority;
    private Date kanDeadline;
    private SubTask subTask;
    private String userEmail;
    public String getProjectName() {
        return projectName;
    }
    public void setProjectName(String projectName) {
        this.projectName = projectName;
    }
    public String getKanTaskName() {
        return kanTaskName;
    }
    public void setKanTaskName(String kanTaskName) {
```

```
        this.kanTaskName = kanTaskName;
    }
    public int getKanTaskId() {
        return kanTaskId;
    }
    public void setKanTaskId(int kanTaskId) {
        this.kanTaskId = kanTaskId;
    }
    public String getAssignee() {
        return assignee;
    }
    public void setAssignee(String assignee) {
        this.assignee = assignee;
    }
    public String getKanPriority() {
        return kanPriority;
    }
    public void setKanPriority(String kanPriority) {
        this.kanPriority = kanPriority;
    }
    public Date getKanDeadline() {
        return kanDeadline;
    }
    public void setKanDeadline(Date kanDeadline) {
        this.kanDeadline = kanDeadline;
    }
    public SubTask getSubTask() {
        return subTask;
    }
    public void setSubTask(SubTask subTask) {
        this.subTask = subTask;
    }
    public String getUserEmail() {
        return userEmail;
    }
    public void setUserEmail(String userEmail) {
        this.userEmail = userEmail;
    }

}
```

SubTask.java

```
public class SubTask {
    private int subTaskId;
    private String subTaskTitle;
    public int getSubTaskId() {
        return subTaskId;
    }
    public void setSubTaskId(int subTaskId) {
        this.subTaskId = subTaskId;
    }
    public String getSubTaskTitle() {
        return subTaskTitle;
    }
```

```
	public void setSubTaskTitle(String subTaskTitle) {
		this.subTaskTitle = subTaskTitle;
	}

}
```

TaskId.aj

```
public aspect TaskId {
	// TODO Auto-generated aspect
	pointcut createGantterTask():call(void createTask(..));
	pointcut updateGantterTask():call(void updateTask(..));

	after(): createGantterTask() {
		System.out.println("Welcome to Gantter!");
		int j=1000;
		Object[] args = thisJoinPoint.getArgs();
//		String gantter = args[0].toString();
		GantterTask gantterTask = (GantterTask)args[0];
		gantterTask.setTaskId(j);
		j++;
	}
	after(): updateGantterTask() {
		Object[] args = thisJoinPoint.getArgs();
		GantterTask gantterTask = (GantterTask)args[0];
		System.out.println("Edit the task : " + gantterTask.getTaskName());

	}
}
```

TaskName.aj

```
import java.util.Scanner;

public aspect TaskName {
	// TODO Auto-generated aspect
	pointcut createGantterTask():call(void createTask(..));
	pointcut updateGantterTask():call(void updateTask(..));

	after(): createGantterTask() {
		System.out.println("Enter the Task name :");
		Object[] args = thisJoinPoint.getArgs();
//		String gantter = args[0].toString();
		GantterTask gantterTask = (GantterTask)args[0];
		Scanner scanner = new Scanner(System.in);
		String taskName = scanner.next();
		gantterTask.setTaskName(taskName);
	}
	after(): updateGantterTask() {
		Object[] args = thisJoinPoint.getArgs();
		GantterTask gantterTask = (GantterTask)args[0];
		System.out.println("Edit the task : " + gantterTask.getTaskName());
		System.out.println("Task name is : "+ gantterTask.getTaskName());
		System.out.println("Enter new task name : ");
		Scanner scanner = new Scanner(System.in);
```

```
                String taskName = scanner.next();
                if(taskName != null)
                        gantterTask.setTaskName(taskName);

        }
}
```

PercentComplete.aj

```
import java.util.Scanner;
public aspect PercentComplete {
        // TODO Auto-generated aspect
        pointcut createGantterTask():call(void createTask(..));
        pointcut updateGantterTask():call(void updateTask(..));

        after(): createGantterTask() {
//              System.out.println("Enter the End date :");
//              System.out.println("Enter the Start Date");
                Object[] args = thisJoinPoint.getArgs();
                GantterTask gantterTask = (GantterTask)args[0];
                Scanner scanner = new Scanner(System.in);
                String percentComplete = scanner.next();
                float percent = Float.valueOf(percentComplete);
                gantterTask.setPercentComplete(percent);
        }

        after(): updateGantterTask() {
                Object[] args = thisJoinPoint.getArgs();
                GantterTask gantterTask = (GantterTask)args[0];
                System.out.println("% complete is : "+ gantterTask.getTaskName());
                System.out.println("Enter new % complete : ");
                Scanner scanner = new Scanner(System.in);
                String percentComplete = scanner.next();
                float percent = Float.valueOf(percentComplete);
                gantterTask.setPercentComplete(percent);
        }
}
```

GanDuration.aj

```
//package G;
import java.text.ParseException;
import java.text.SimpleDateFormat;
import java.util.Date;
import java.util.Scanner;

public aspect GanDuration {
        // TODO Auto-generated aspect
        pointcut createGantterTask():call(void createTask(..));
        pointcut updateGantterTask():call(void updateTask(..));

        after(): createGantterTask() {
                Object[] args = thisJoinPoint.getArgs();
                GantterTask gantterTask = (GantterTask)args[0];
                System.out.println("Enter the Start Date in mm/dd/yyyy");
                Scanner scanner = new Scanner(System.in);
                String startDate = scanner.next();
```

```
		System.out.println("Enter the End date in mm/dd/yyyy :");
		String endDate = scanner.next();

		SimpleDateFormat format = new SimpleDateFormat("MM/dd/yyyy");
		Date start_date = null;
		Date end_date = null;
		try {
			start_date = format.parse(startDate);
			end_date = format.parse(endDate);
		} catch (Exception e) {
			// TODO Auto-generated catch block
			e.printStackTrace();
		}
		long diff = end_date.getTime() - start_date.getTime();
		long diffDays = diff / (24 * 60 * 60 * 1000);
		int duration = (int) diffDays;
		gantterTask.setgDuration(duration);
		gantterTask.setStartdate(start_date);
		gantterTask.setEndDate(end_date);
}

after(): updateGantterTask() {
		Object[] args = thisJoinPoint.getArgs();
		GantterTask gantterTask = (GantterTask)args[0];
		System.out.println("start date is : "+ gantterTask.getStartdate());
		System.out.println("end date is : "+ gantterTask.getEndDate());
		System.out.println("Enter new start date in mm/dd/yyyy : ");
		Scanner scanner = new Scanner(System.in);
		String startDate = scanner.next();

		System.out.println("Enter new end date in mm/dd/yyyy : ");
		String endDate = scanner.next();

		SimpleDateFormat format = new SimpleDateFormat("MM/dd/yyyy");
		Date start_date = null;
		Date end_date = null;

			try {
				start_date = format.parse(startDate);
			} catch (ParseException e) {
				// TODO Auto-generated catch block
				e.printStackTrace();
			}
			try {
				end_date = format.parse(endDate);
			} catch (ParseException e) {
				// TODO Auto-generated catch block
				e.printStackTrace();
			}


		long diff = end_date.getTime() - start_date.getTime();
		long diffDays = diff / (24 * 60 * 60 * 1000);
		int duration = (int) diffDays;
```

```
                if(duration != 0)
                        gantterTask.setgDuration(duration);
                if(startDate != null)
                        gantterTask.setStartdate(start_date);
                if(endDate != null)
                        gantterTask.setEndDate(end_date);
        }
}
```

GanTaskDeadline.aj

```
// package GA;

import java.text.ParseException;
import java.text.SimpleDateFormat;
import java.util.Date;
import java.util.Scanner;


public aspect GanTaskDeadline {
        pointcut createGantterTask():call(void createTask(..));
        pointcut updateGantterTask():call(void updateTask(..));

        after(): createGantterTask() {
                Object[] args = thisJoinPoint.getArgs();
                GantterTask gantterTask = (GantterTask)args[0];
                System.out.println("Enter the deadline");
                Scanner scanner = new Scanner(System.in);
                String dead_Line = scanner.next();
                SimpleDateFormat format = new SimpleDateFormat("MM/dd/yyyy");
                Date deadLine = null;

                        try {
                                deadLine = format.parse(dead_Line);
                        } catch (Exception e) {
                                // TODO Auto-generated catch block
                                e.printStackTrace();
                        }

                gantterTask.setgDeadLine(deadLine);
        }

        after(): updateGantterTask() {
                Object[] args = thisJoinPoint.getArgs();
                GantterTask gantterTask = (GantterTask)args[0];
                System.out.println("The deadline is : "+gantterTask.getgDeadLine());
                System.out.println("Enter the deadline");
                Scanner scanner = new Scanner(System.in);
                String dead_Line = scanner.next();
                SimpleDateFormat format = new SimpleDateFormat("MM/dd/yyyy");
                Date deadLine = null;

                        try {
                                deadLine = format.parse(dead_Line);
                        } catch (ParseException e) {
```

```
                        // TODO Auto-generated catch block
                        e.printStackTrace();
                }

                if(dead_Line != null)
                        gantterTask.setgDeadLine(deadLine);
        }
}
```

ResourceName.aj

```
import java.util.Scanner;
public aspect ResourceName {
        // TODO Auto-generated aspect
        pointcut createGantterTask():call(void createTask(..));
        pointcut updateGantterTask():call(void updateTask(..));

        after(): createGantterTask() {
                System.out.println("Do you want to add Resources? (y/n)");
                Scanner scanner = new Scanner(System.in);
                String yesNo = scanner.next();
                int j = 10000;
                if(yesNo == "y") {
                        Object[] args = thisJoinPoint.getArgs();
                        GantterTask gantterTask = (GantterTask)args[0];
                        gantterTask.getResource().setResourceId(j);
                        System.out.println("enter resource name");
                        String resourceName = scanner.next();
                        if(resourceName != null)
                                gantterTask.getResource().setResourceName(resourceName);
                }
                else if(yesNo == "n") {
                        return;
                }
        }

        after(): updateGantterTask() {
                System.out.println("Do you want to add Resources? (y/n)");
                Scanner scanner = new Scanner(System.in);
                String yesNo = scanner.next();
                int j = 10000;
                if(yesNo == "y") {
                        Object[] args = thisJoinPoint.getArgs();
                        GantterTask gantterTask = (GantterTask)args[0];
                        gantterTask.getResource().setResourceId(j);
                        System.out.println("enter resource name");
                        String resourceName = scanner.next();
                        if(resourceName != null)
                                gantterTask.getResource().setResourceName(resourceName);
                }
                else if(yesNo == "n") {
                        return;
                }
        }}
```

RiskName.aj

```
import java.util.Scanner;
public aspect RiskName {
	// TODO Auto-generated aspect
	pointcut createGantterTask():call(void createTask(..));
	pointcut updateGantterTask():call(void updateTask(..));

	after(): createGantterTask() {
		System.out.println("Do you want to add Risk? (y/n)");
		Scanner scanner = new Scanner(System.in);
		String yesNo = scanner.next();
		int j = 100;
		if(yesNo == "y") {
			Object[] args = thisJoinPoint.getArgs();
			GantterTask gantterTask = (GantterTask)args[0];
			gantterTask.getRisk().setRiskId(j);
			System.out.println("enter resource name");
			String riskName = scanner.next();
			if(riskName != null)
				gantterTask.getRisk().setRiskName(riskName);
			j++;
		}
		else if(yesNo == "n") {
			return;
		}
	}

	after(): updateGantterTask() {
		System.out.println("Do you want to add Resources? (y/n)");
		Scanner scanner = new Scanner(System.in);
		String yesNo = scanner.next();
		int j = 100;
		if(yesNo == "y") {
			Object[] args = thisJoinPoint.getArgs();
			GantterTask gantterTask = (GantterTask)args[0];
			gantterTask.getRisk().setRiskId(j);
			System.out.println("enter resource name");
			String riskName = scanner.next();
			if(riskName != null)
				gantterTask.getRisk().setRiskName(riskName);
			j++;
		}
		else if(yesNo == "n") {
			return;
		}
	}
}
```

BacklogId.aj

```
public aspect BacklogId {
	// TODO Auto-generated aspect
	pointcut createBacklogId():call(void createBackLog(..));

	after(): createBacklogId() {
```

```
                System.out.println("wlecome to EasyBacklog! ");
                int j=2000;
                Object[] args = thisJoinPoint.getArgs();
                EasyBacklog backLog = (EasyBacklog)args[0];
                backLog.setBacklogId(j);
                j++;

        }
}
```

BacklogName.aj

```
import java.util.Scanner;
public aspect BacklogName {
        // TODO Auto-generated aspect
        pointcut createBacklogName():call(void createBackLog(..));

        after(): createBacklogName() {
                System.out.println("Enter the Backlog name :");
                Object[] args = thisJoinPoint.getArgs();
                EasyBacklog backLog = (EasyBacklog)args[0];
                Scanner scanner = new Scanner(System.in);
                String backlogName = scanner.next();
                backLog.setBacklogName(backlogName);
        }
}
```

Story.aj

```
import java.util.Scanner;
import java.util.Scanner;
public aspect Story {
        // TODO Auto-generated aspect
        pointcut createStory():call(void createBackLog(..));
        pointcut updateStory():call(void updateBackLog(..));
                after(): createStory() {
                        System.out.println("Enter Story :As a User ,I want to, So that....");
                        Object[] args = thisJoinPoint.getArgs();
                        EasyBacklog backLog = (EasyBacklog)args[0];
                        Scanner scanner = new Scanner(System.in);
                        String story = scanner.next();
                        backLog.setStory(story);
                }
                after():updateStory(){
                        System.out.println("do you want to update Y/N");
                        Scanner s=new Scanner(System.in);
                   String reply=s.next();
                   if(reply.equalsIgnoreCase("y")){
                        Object[] args = thisJoinPoint.getArgs();
                                EasyBacklog backLog = (EasyBacklog)args[0];
                                Scanner scanner = new Scanner(System.in);
                                String story = scanner.next();
                                System.out.println("story updated to");
                                backLog.setStory(story);
                   }else{
```

```
                        return;
                    }
                }
}
```

Acceptance.aj

```
import java.util.Scanner;

public aspect Acceptance {
        // TODO Auto-generated aspect
        pointcut createAcceptance():call(void createBackLog(..));
        pointcut updateAcceptance():call(void updateBackLog(..));

                after(): createAcceptance() {
                        System.out.println("Enter Acceptance criteria :");
                        Object[] args = thisJoinPoint.getArgs();
                        EasyBacklog backLog = (EasyBacklog)args[0];
                        Scanner scanner = new Scanner(System.in);
                        String acceptance = scanner.next();
                        backLog.setAcceptance(acceptance);
                }

                after():updateAcceptance(){
                        System.out.println("do you want to update Y/N");
                        Scanner s=new Scanner(System.in);
                   String reply=s.next();
                   if(reply.equalsIgnoreCase("y")){
                        Object[] args = thisJoinPoint.getArgs();
                                EasyBacklog backLog = (EasyBacklog)args[0];
                                Scanner scanner = new Scanner(System.in);
                                String acceptance = scanner.next();
                                System.out.println("acceptance updated to");
                                backLog.setAcceptance(acceptance);
                   }else{
                        return;
                   }
                }
}
```

Duration.aj

```
import java.util.Scanner;
public aspect Duration {
        // TODO Auto-generated aspect
pointcut createDuration():call(void createBackLog(..));

        after(): createDuration() {
                System.out.println("Enter the Duration :");
                Object[] args = thisJoinPoint.getArgs();
                EasyBacklog backLog = (EasyBacklog)args[0];
                Scanner scanner = new Scanner(System.in);
                String duration = scanner.next();
                int d=Integer.parseInt(duration);
                backLog.getSprint().setDuration(d);
```

```
	}
}
```

VelPerPerson.aj

```
import java.util.Scanner;

public aspect VelPerPerson {
	// TODO Auto-generated aspect
	pointcut createVelPerPerson():call(void createBackLog(..));

	after(): createVelPerPerson() {
		System.out.println("Calc velocity :");
		Object[] args = thisJoinPoint.getArgs();
		EasyBacklog backLog = (EasyBacklog)args[0];
		System.out.println("enter team size");
		Scanner scanner = new Scanner(System.in);
		String teamSize = scanner.next();
		int ts=Integer.parseInt(teamSize);
		int velocity=ts*30;
		backLog.getSprint().setVelPerPerson(velPerPerson);
	}
}
```

SprintName.aj

```
import  java.util.Scanner;
public aspect SprintName {
	// TODO Auto-generated aspect
	pointcut createSprintName():call(void createBackLog(..));

	after(): createSprintName() {
		System.out.println("Enter the sprint name :");
		Object[] args = thisJoinPoint.getArgs();
		EasyBacklog backLog = (EasyBacklog)args[0];
		Scanner scanner = new Scanner(System.in);
		String sprintName = scanner.next();
		backLog.getSprint().setSprintName(sprintName);
	}
}
```

AvgVelocityDay.aj

```
import java.util.Scanner;
public aspect AvgVelocityDay {
	// TODO Auto-generated aspect
	pointcut createAvgVelocity():call(void createBackLog(..));
	pointcut updateAvgVelocity():call(void updateBackLog(..));

		after(): createAvgVelocity() {
			System.out.println("Calc avg velocity per day :");
			Object[] args = thisJoinPoint.getArgs();
			EasyBacklog backLog = (EasyBacklog)args[0];
			System.out.println("velocity : 3");
		int velocity=3;
```

```
                        System.out.println("Rate: 800");
                        int rate=800;
                        System.out.println("enter points:");
                        Scanner scanner = new Scanner(System.in);
                        String points = scanner.next();
                        int p=Integer.parseInt(points);
                        int cost=(rate/velocity)*p;
                        System.out.println("cost : "+cost);
                        float avgVelPerDay=p/velocity;
                        backLog.setAvgVelPerDay(avgVelPerDay);
                }

                after():updateAvgVelocity(){
                        System.out.println("do you want to update Y/N");
                        Scanner s=new Scanner(System.in);
                  String reply=s.next();
                  if(reply.equalsIgnoreCase("y")){
                        Object[] args = thisJoinPoint.getArgs();
                                EasyBacklog backLog = (EasyBacklog)args[0];
                                System.out.println("velocity : 3");
                        int velocity=3;
                                System.out.println("Rate: 800");
                                int rate=800;
                                System.out.println("enter points:");
                                Scanner scanner = new Scanner(System.in);
                                String points = scanner.next();
                                int p=Integer.parseInt(points);
                                int cost=(rate/velocity)*p;
                                System.out.println("cost : "+cost);
                                float avgVelPerDay=p/velocity;
                                backLog.setAvgVelPerDay(avgVelPerDay);
                  }else{
                        return;
                  }
                }
}
```

Project.aj

```
import java.util.Scanner;

public aspect Project {
        // TODO Auto-generated aspect
        pointcut createProject():call(void createKanbanTask(..));
        pointcut updateBackLog():call(void updateBackLog(..));

        after(): createProject() {
                System.out.println("Enter project name :");
                Object[] args = thisJoinPoint.getArgs();
                KanbanTask kanbantask = (KanbanTask)args[0];
                Scanner scanner = new Scanner(System.in);
                String projectName = scanner.next();
                kanbantask.setProjectName(projectName);
        }
```

```
	after(): updateBackLog() {

		Object[] args = thisJoinPoint.getArgs();
		KanbanTask kanbantask = (KanbanTask)args[0];
		System.out.println("project name :"+kanbantask.getProjectName());
		System.out.println("Enter project name :");
		Scanner scanner = new Scanner(System.in);
		String projectName = scanner.next();
		if(projectName != null)
		kanbantask.setProjectName(projectName);
	}
}
```

TaskTitle.aj

```
import java.util.Scanner;

public aspect TaskTitle {
	// TODO Auto-generated aspect
	pointcut createTaskTitle():call(void createKanbanTask(..));
	pointcut updateBackLog():call(void updateBackLog(..));

	after(): createTaskTitle() {
		System.out.println("Enter task title");
		Object[] args = thisJoinPoint.getArgs();
		KanbanTask kanbantask = (KanbanTask)args[0];
		Scanner scanner = new Scanner(System.in);
		String kanTaskName = scanner.next();
		kanbantask.setKanTaskName(kanTaskName);
	}

	after(): updateBackLog() {

		Object[] args = thisJoinPoint.getArgs();
		KanbanTask kanbantask = (KanbanTask)args[0];
		System.out.println("task title : "+kanbantask.getKanTaskName());
		System.out.println("Enter task title");
		Scanner scanner = new Scanner(System.in);
		String kanTaskName = scanner.next();
		if(kanTaskName != null)
		kanbantask.setKanTaskName(kanTaskName);
	}
}
```

TaskUId.aj

```
import java.util.Scanner;

public aspect TaskUId {
	// TODO Auto-generated aspect
	pointcut createTaskUId():call(void createKanbanTask(..));

	after(): createTaskUId() {
		System.out.println("welcome to kanban");
		int kanTaskId=4000;
```

```
                Object[] args = thisJoinPoint.getArgs();
                KanbanTask kanbantask = (KanbanTask)args[0];
                kanbantask.setKanTaskId(kanTaskId);
                kanTaskId++;
        }
}
```

Assignee.aj

```
import java.util.Scanner;

public aspect Assignee {
        // TODO Auto-generated aspect
        pointcut createAssignee():call(void createKanbanTask(..));
        pointcut updateBackLog():call(void updateBackLog(..));

        after(): createAssignee() {
                System.out.println("Enter Assignee");
                Object[] args = thisJoinPoint.getArgs();
                KanbanTask kanbantask = (KanbanTask)args[0];
                Scanner scanner = new Scanner(System.in);
                String assignee = scanner.next();
                kanbantask.setAssignee(assignee);
        }

        after(): updateBackLog() {

                Object[] args = thisJoinPoint.getArgs();
                KanbanTask kanbantask = (KanbanTask)args[0];
                System.out.println("Assignee is:"+kanbantask.getAssignee());
                System.out.println("Enter Assignee : ");
                Scanner scanner = new Scanner(System.in);
                String assignee = scanner.next();
                if(assignee != null)
                        kanbantask.setAssignee(assignee);
        }
}
```

Priority.aj

```
import java.util.Scanner;

public aspect Priority {
        // TODO Auto-generated aspect
        pointcut createPriority():call(void createKanbanTask(..));
        pointcut updateBackLog():call(void updateBackLog(..));

        after(): createPriority() {
                System.out.println("Enter priority LOW, AVERAGE,HIGH");
                Object[] args = thisJoinPoint.getArgs();
                KanbanTask kanbantask = (KanbanTask)args[0];
                Scanner scanner = new Scanner(System.in);
                String kanPriority = scanner.next();
                kanbantask.setKanPriority(kanPriority);
        }
```

```
	after(): updateBackLog() {

		Object[] args = thisJoinPoint.getArgs();
		KanbanTask kanbantask = (KanbanTask)args[0];
		System.out.println("priority is : "+kanbantask.getKanPriority());
		System.out.println("Enter priority LOW, AVERAGE,HIGH");
		Scanner scanner = new Scanner(System.in);
		String kanPriority = scanner.next();
		if(kanPriority != null)
		kanbantask.setKanPriority(kanPriority);
	}
}
```

Deadline.aj

```
import java.text.DateFormat;
import java.text.SimpleDateFormat;
import java.util.Date;
import java.util.Scanner;

public aspect Deadline {
	// TODO Auto-generated aspect
	pointcut createDeadline():call(void createKanbanTask(..));
	pointcut updateBackLog():call(void updateBackLog(..));

	after(): createDeadline() {
		System.out.println("Enter deadline: yyyy-MM-dd format");
		Object[] args = thisJoinPoint.getArgs();
		KanbanTask kanbantask = (KanbanTask)args[0];
		Scanner scanner = new Scanner(System.in);
		String deadline = scanner.next();
		DateFormat dateFormat = new SimpleDateFormat("yyyy-MM-dd");
		Date kanDeadline = dateFormat.parse(deadline);
		kanbantask.setKanDeadline(kanDeadline);
	}

	after(): updateBackLog() {

		Object[] args = thisJoinPoint.getArgs();
		KanbanTask kanbantask = (KanbanTask)args[0];
		System.out.println("deadline is : "+kanbantask.getKanDeadline());
		System.out.println("Enter deadline: yyyy-MM-dd format");
		Scanner scanner = new Scanner(System.in);
		String deadline = scanner.next();
		DateFormat dateFormat = new SimpleDateFormat("yyyy-MM-dd");
		Date kanDeadline = dateFormat.parse(deadline);
		if(deadline != null)
		kanbantask.setKanDeadline(kanDeadline);
	}
}
```

SubTaskId.aj

```
import java.util.Scanner;

public aspect SubTaskId {
	// TODO Auto-generated aspect
	pointcut createSubTaskId():call(void createKanbanTask(..));

	after(): createSubTaskId() {
		System.out.println("You can create subtasks to task");
		int subTaskId=3000;
		Object[] args = thisJoinPoint.getArgs();
		KanbanTask kanbantask = (KanbanTask)args[0];
		kanbantask.getSubTask().setSubTaskId(subTaskId);
		subTaskId++;
	}
}
```

SubTaskTitle.aj

```
import java.util.Scanner;

public aspect SubTaskTitle {
	// TODO Auto-generated aspect
	pointcut createSubTasktitle():call(void createKanbanTask(..));
	pointcut updateBackLog():call(void updateBackLog(..));

	after(): createSubTasktitle() {
		System.out.println("Enter sub task title :");
		Object[] args = thisJoinPoint.getArgs();
		KanbanTask kanbantask = (KanbanTask)args[0];
		Scanner scanner = new Scanner(System.in);
		String subTaskTitle = scanner.next();
		kanbantask.getSubTask().setSubTaskTitle(subTaskTitle);
	}

	after(): updateBackLog() {

		Object[] args = thisJoinPoint.getArgs();
		KanbanTask kanbantask = (KanbanTask)args[0];
		System.out.println("sub task title :"+kanbantask.getSubTask().getSubTaskTitle());
		System.out.println("Enter sub task title :");
		Scanner scanner = new Scanner(System.in);
		String subTaskTitle = scanner.next();
		if(subTaskTitle != null)
		kanbantask.getSubTask().setSubTaskTitle(subTaskTitle);
	}
}
```

userEmail.aj

```
import java.util.Scanner;

public aspect userEmail {
	// TODO Auto-generated aspect
	pointcut createuserEmail():call(void createKanbanTask(..));
```

```
        pointcut updateBackLog():call(void updateBackLog(..));

        after(): createuserEmail() {
                System.out.println("enter user email abc@xyz.com");
                Object[] args = thisJoinPoint.getArgs();
                KanbanTask kanbantask = (KanbanTask)args[0];
           Scanner s=new Scanner(System.in);
           String userEmail=s.next();
                kanbantask.setUserEmail(userEmail);
        }

        after(): updateBackLog() {

                Object[] args = thisJoinPoint.getArgs();
                KanbanTask kanbantask = (KanbanTask)args[0];
                System.out.println("user email : "+kanbantask.getUserEmail());
                System.out.println("enter user email abc@xyz.com");
           Scanner s=new Scanner(System.in);
           String userEmail=s.next();
           if(userEmail != null)
                kanbantask.setUserEmail(userEmail);
        }
}
```