CanvasPath

Eric Roeum

March 1, 2019

Contents

List of Figures

1 Introduction

At Lion State University, staff and students need a system in order to maintain information about classes, students, grades, roles, and other aspects of school. Because of this, a database is required to hold all of this information in a usable format that student and teachers can access both easily and securely. This system will be called CanvasPath. The CanvasPath application will primarily be composed of four components:

- A login system
- Viewing courses
- Scheduling courses
- Deriving student infromation

As such, this system will be web browser based with a client-server implemented model. The client system will only be used for input/output, while the server will handle all database operations.

This system will be an updated version of the current applications: Canvas and LionPath. By integrating both systems into one, users will have more ease of use and will be able to do operations in one system instead of having to switch between both.

1.1 Objective

The purpose of the document is to propose an implmentation for the CanvasPath application along with potential features. In addition to doing so, this document contains how the internal database should be organized along with potential designs. As a result, this document is divided primarily into five components:

- Requirements analysis
- Web Application Design
- Conceptual database design
- Technological survey
- Logical database design

1.2 Stylization

Within this document, certain styles and notations will be used for readability. More specifically, italicization is resevered for only variable names (i.e. *Height*). Any other forms of text can be used in any other context. For entity-relationship (ER) diagrams, the textbook notation is used (See reference [1]).

2 Requirements Analysis

This section goes over the requirements proposed by LionState University for the CanvasPath project.

2.1 Overview

Because CanvasPath is intended to replace current course managements systems at LionState University, a lot of the requirements come from updating the features of the current system, Canvas and LionPath. As such, CanvasPath requires a login system for the user to be identified, a system to control course scheduling/viewing, and accessing personal info primarily. As a result, each system must be implemented in order to meet the requirements of the current application. Of course, it is important to include new features as well. However, current features need to be implemented before new features should be developed.

2.2 System Overview

In terms of easibility, Canvas has the advantage over Lionpath, so the overall system should mimic the Canvas system more so. As a result, the canvas design flow will be followed. When the user first opens the application, for example, they are presented with a login screen such that they can identify who they are without allowing others to access the same information. After logging in, the system should detect if the user is a student, faculty member, or administrator.

Each user will be presented with three levels of usability:

- A student can view and sign-up for classes
- A faculty member can view, sign-up, and create classes/groups
- An administrator can do everything a student and faculty member can do, but also create new users.

Another aspect that canvas has is viewing individual course details. Within each course, students and faculty members should be able to view grades, announcments, assignmennts, files, grades, associated people, and view syllabi.

Within LionPath, features unique to LionPath should also be added to this Canvas-like system. For example, scheduling course, viewing available courses, prerequisite courses, academic records, tuition, and research activities. As a result, after logging into the system, a user should be given a menu to choose the correct action.

2.3 Users

There are two primary users: a student and a faculty member, with the special user called the administrator. Each category is not necessarily disjoint from each other though. For example, a student may also be a faculty member. Both the student and faculty member should have personal information stored as well as courses taken/taught, grades, etc. The primary difference between the two is that a student should take a class whereas a faculty member should support a class.

A student is any person who is currently taking or has taken a class. Because of this, each student requires past courses, current courses, future courses, grades, assignments, and files to be stored in order to meet this requirement academically. On more personal levels, it is also important to store more personal information. For example, a student's primary address, personal email, school email ID, passwords, emergency contacts are essential to keep. Further relations need to be stored as well. For example, the instructors, courses, sections, departments, etc. that the student is related to should alos be stored.

On the other hand, any person who is currently supporting or has supported a class is a factuly member.

This includes, teachers, professors, researchers, teaching assistants, learning assistants, and more. Faculty members are very similar in what data is required on them: past courses, current courses, future courses, files, research, budget, etc. More information is then required for human resourse (HR) purposes: pay-rate, years worked, department.

The final administrator user is particularly useful for implementing new groups and users. These users will have full control to the system and are able to both access and manipulate any data.

2.3.1 Student

A student is any user who is enrolled at Lion State University. It is important to maintain information about enrolled students so both students and faculty members can enroll into courses appropriately. Information stored about students should include:

- Name: Student Name
- ID: Student ID
- Age: Student Age
- Gender: Male/Female/Other
- Email Address: Personal email address of student
- Home Address: Permananent/Mailing address of student
- Login Password: Password to log into system (Encoded into SHA256)
- Majors/Minors: Completed/Intended graudation path of student

2.3.2 Faculty Member

A faculty member is any user who supports any course at Lion State University. Note that courses may not necessarily be purely academic and supporting a course does not necessarily mean teaching a class. Still, it is importannt to maintain information about active faculty member so that they can properly support their respective course. Information stored about faculty members include the following:

- Name: Faculty Name
- ID: Faculty ID (Note that this is synonomous for Student ID as they come from the same pool of values)
- Age: Faculty Age
- Gender: Male/Female/Other
- Email Address: Personal email address of faculty member
- Home Address: Permananent/Mailing address of faculty member
- Login Password: Password to log into system (Also encoded into SHA256)
- Department: Department that faculty corresponds
- Pay-grade: Level of pay faculty member gets paid
- Semesters Working: Amount of years worked by faculty member
- Office Address: Working office of faculty member
- Title: Title of faculty member

2.4 Groups

In order to avoid duplication of data, groups are made to rerpresent areas where a group of individuals of users share a certain aspsect. These groups are primarily categorized by the following:

1. Courses: a class which a student may take and a faculty member may teach/support. Courses can only be offered by one department (though duplicate courses can have different departments i.e. CS/DS 200). Within each course, the course name, course ID, department, sections, primary teacher, support staff, and students all need to be stored.

2. Deparments: The certain degree program that classes identify with. For example computer science, psychology, and engineering are considered as departments. Note that duplicate departments cannot exist. Within the department group, information regarding faculty members are mentioned as well as majors, minors, names of programs, etc. The departments group individual courses by specific major/minor programs.

3. Section: a subset of a course. Because courses can be very large, it is important to maintain smaller groups in order to provide multiple sections of the same class. Further, sections allow different course material. For example, a capstone section can now be implemented in order to have different assignments.

2.4.1 Courses

A course represents, not only a class that a student can take, but also an activity that some students or facilty members may participate in. For example, a research activity can be considered a course, though it may not necessarily be for credit or have the ability enroll students. However, using the research activity example, a single course will be implemented to represent all research activities on campus while each individual research activity is distibuted bia sections within the course information.

2.4.2 Departments

A department is a certain group which courses can associatite with, more specifically by majors. For example, computer science, psychology, math, physics are all appropriate departments, though none of them have to necessarily be degree program offered by the university. Further, administration may be able to control department budgets and activities of courses thorugh a single entity instead of controling each individaul activity.

2.4.3 Section

A section is a subset of a single course that a student can enroll. Usually, sections of the same course vary in time, but not necessarily. Further a section specifies the type of the course. For example sections can represent the following:

- Regular: A class where students physically attend the course
- Online: A class where student access course information electronically
- Hybrid: A mixture of both regular and online classes
- Research: A research opportunity that students/faculty members can participate
- Internship: An opportunity that students can take off school during a semester term.
- Coop: An opportunity that students can take off school during multiple semester terms.

3 Web Application Design

This section will cover the inteded design and implmentation for CanvasPath. The web application will primarily be controlled via a server-client communication protocal, where all client-side operations will occur through a web browser.

3.1 Webpage Overview

As stated previously all communcations will follow the server-client communication protocal. The client-side of this application will be the highlight of this section. Within this aspect, there are four primary windows:

- Login Screen
- Homepage
- Personal Information
- Current Schedule
- Class Scheduler
- Finances

The following diagram shows how each window should interfaces with one another. The first screen any user is presented with is the login screen. Afterwards, any window that the current window is pointing towards can be accessed.

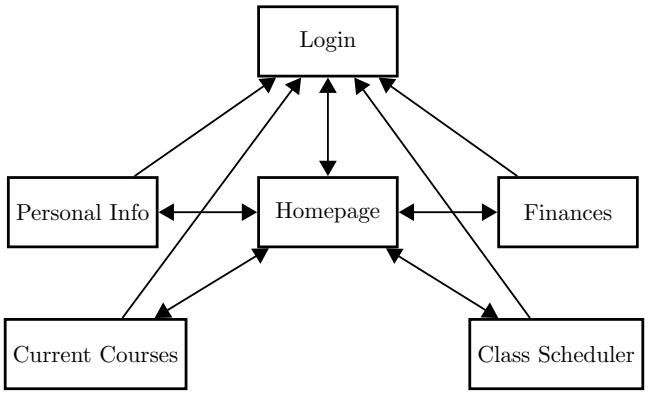

Figure 1: Flow Diagram of Webpage Screens

3.2 Login Screen

The login screen's only purpose is to allow users to identify themselves securily. In order to avoid any confusion, this procedure is kept simple with two textboxes: the user id and password; and a button: submit request. Though currently, the Canvas and LionPath applications have a "Forgot Password" button, this feature will not be added in order to reduce complications and stay in the scope of the assignment.

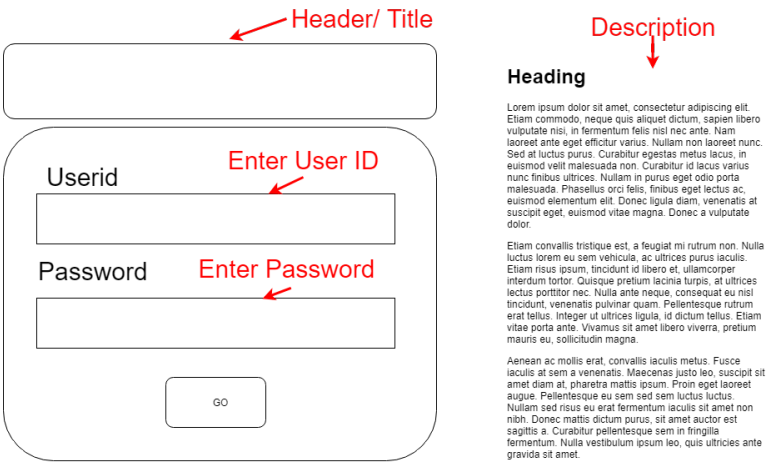

Figure 2: Login Webpage

3.3 Homepage

The homepage is very similar to Canvas's Dashborad page. This page will take the most recent activity for the user and display it on the screen such that it is very easy to view and use the most likely activity. Further the homeage acts as a hub for the rest of the webpages; all webpages can be access from the home webpage.

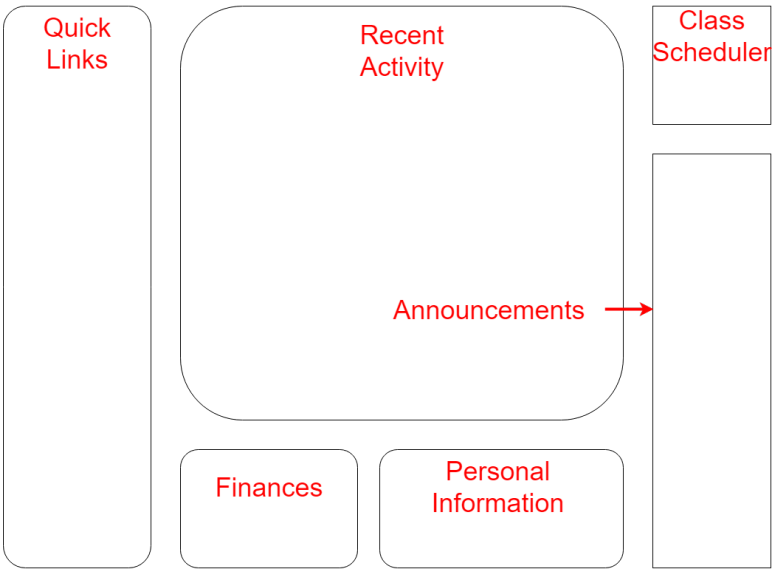

Figure 3: Homepage

3.4 Personal Infromation

The personal information screen allows the user to both access and manipulate any personal data stored by the system. For example, if the user wishes to change his/her password, address, phone number, and others, he/she can do so in the personal information webpage. This page is also very simple: a table shows all the currently stored data. After each row, a button is present that allows the user to change the respective field

by enabling the textbox. While changing information, two new buttons appear in order to either "confirm" or "cancel" the change. On the top of the screen, there exists tabs in order to scroll through possible categories such as academics, finances, contact infromation, etc. This webpage exists primarily as an archival page for older information.

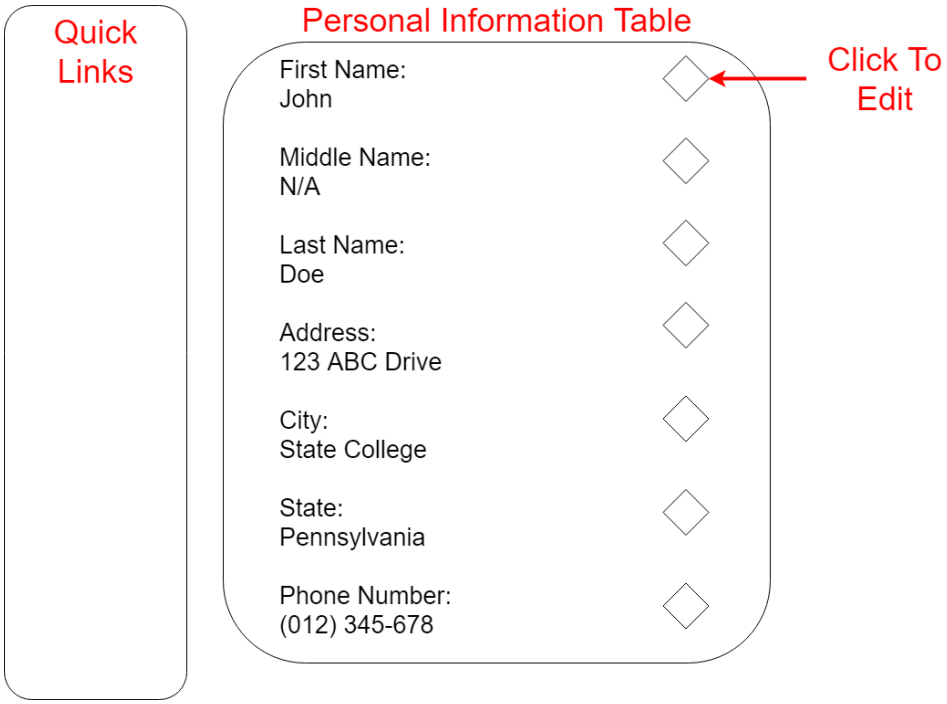

Figure 4: Personal Information Webpage

3.5 Current Schedule

The current schedule provides students and faculty members access to current classes. In this webpages, each student can view all of his/her courses, enter the specific course, and do anything relevant to that class. This is most similar to Canvas's "Courses" tab. Students and faculty members can access enrolled courses and see upcoming assignments, view any announcements, check grades, view files, etc. Faculy members have special priveledes, however. They can, for example, create announcements, upload files, and change grades.

This page has a very simple design. On the left-hand-side, there exists tabs to get to other screen. In the center, a table displays all currently enrolled courses. If a user selects a course in that table, the table disappears and that course's specific course page becomes active.

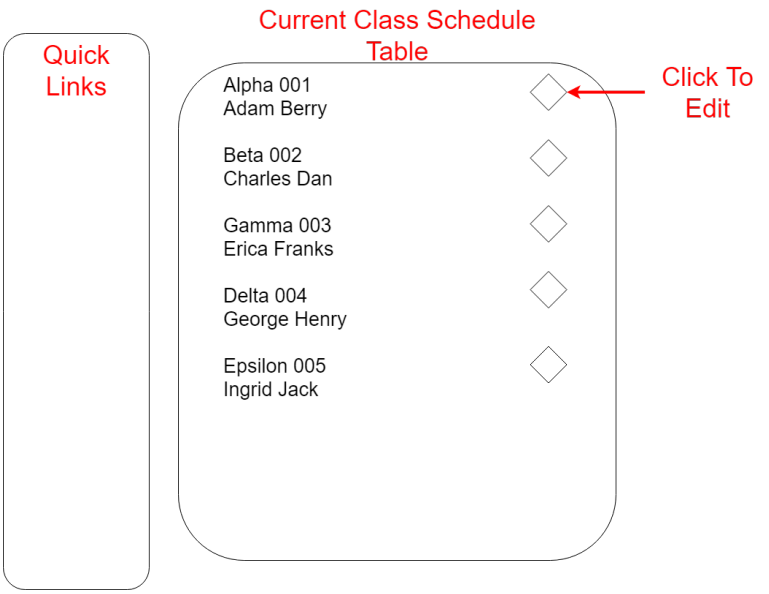

Figure 5: Table of currently enrolled courses

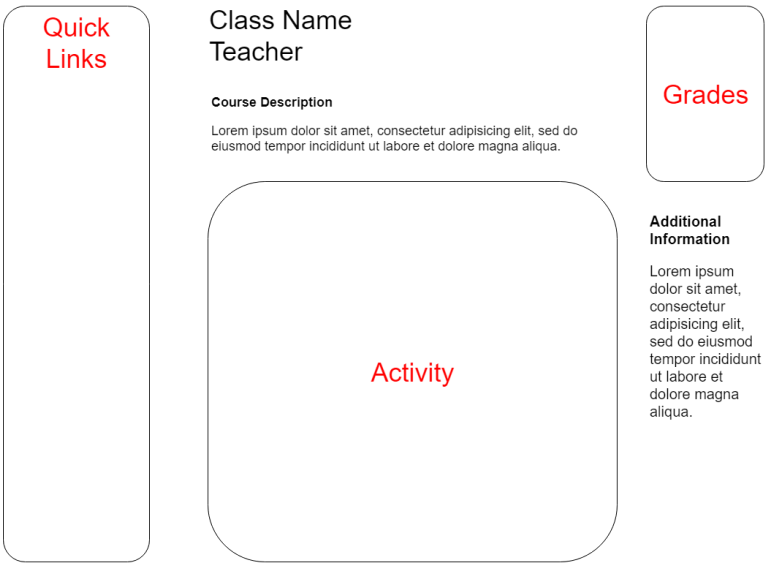

Figure 6: Example of a course page

3.6 Class Scheduler

The class scheduler allows students to schedule for courses for future semesters as well as faculty members to create courses. Any course that a student schedules must be approved by the instructor of the course. Any course that a faculty member creates must be approved by an administrator. This most closely follows LionPath's Scheduler Builder tool, combined with the "Search Classes" screen. Following a similar design, the CanvasPath application will follow a folder system: classes are organized per department per semester. The page is very similar to the current schedule page mentioned in section 3.5.

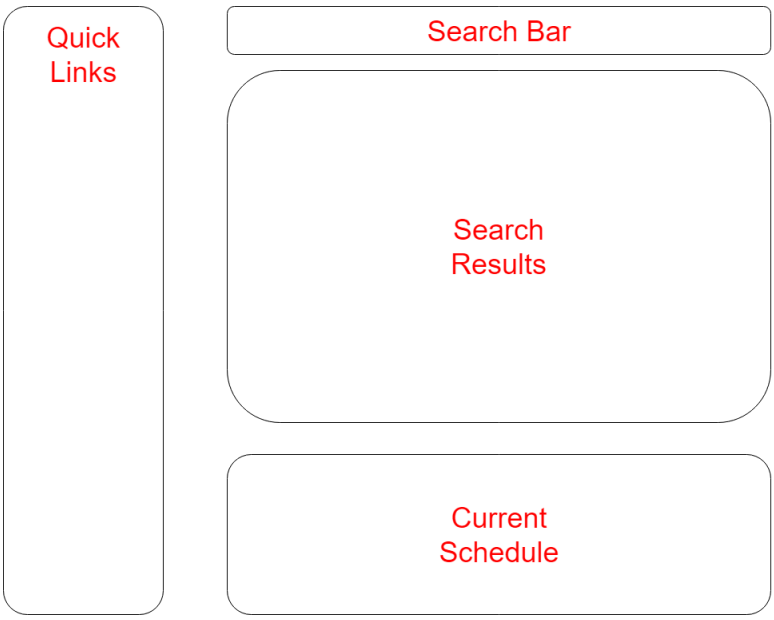

Figure 7: Table of currently enrolled courses

Within this class scheduler, a search bar is at the top of the page in order to search for the class desired. Any results matching that string will be displayed in the search results window, where users can get more information on each specific course. If the user is interested, he/she can add that to his/her current schedule, displayed at the bottom. Any course added, is then requested to the instructor of the course for approval. At the start of the semester, any student who was denied entry to the course or is still waiting is removed from this menu. Note that any courses not added to the current schedule are not saved, so the user must research for the course. A faculty member has the additional feature to add to the course lists. Approval is required by an administrator however.

3.7 Finances

The finance web page displays anything related to money. A student can view estimated tuitions, housing costs, and meal plans. A faculty member has the ability to view his/her paygrade, any bills, past payments, any forms, and payment type. Most of the fields are immutable by anyone except an administrator. The webpage is very simple, with a single table to view the previously stated fields.

For the purposes of this project, no information will be mutable and no payments are actually generated. Doing so would add unnecessary complications and require actual finances. Further, the finance page will not be completely accurate; due to the scope of this project, no pricing details will be implemented and the estimates will be completely fictional.

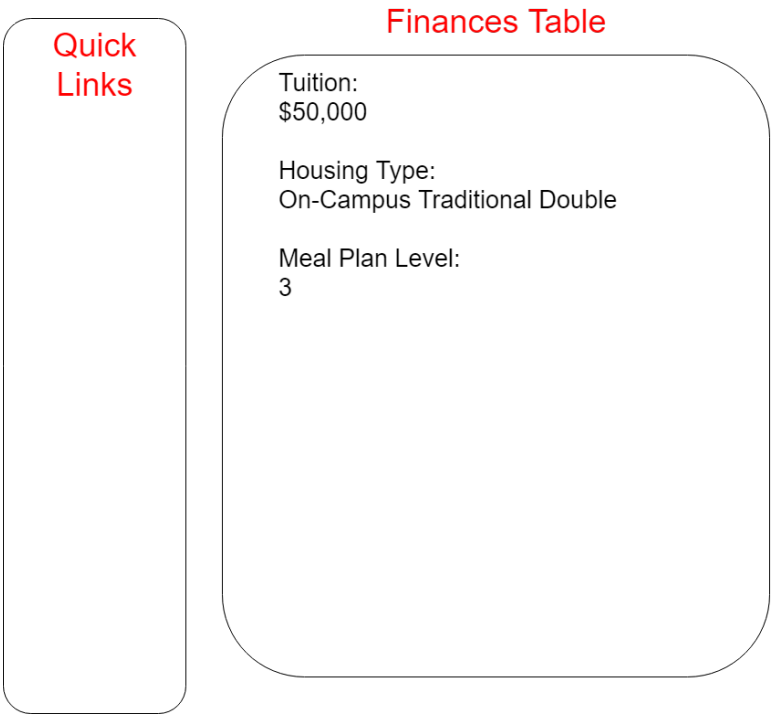

Figure 8: Finances Webpage

4 Conceptual Database Design

This section goes over the database design of the CanvasPath project using primarily ER diagrams. The following figures desribes the primary overall structure of the database. Any attributes have been ommitted for the purposes of avoiding confusion. The main entities are a student, a faculty member, and a section. Note that a section is primary because it is simply a derivation from a course from a department.

There are, however, 15 entitites in this current section:

- *Student*: Any student enrolled in LionState University
- *PartTime_Student*: Any student taking fewer than 12 credits
- *Regular_Student*: Any student taking 12 or more credits
- *Honor_Student*: Any student enrolled in the honors program
- *Inactive_Student*: Any future or past students
- *Graduate_Student*: Any student in a graduate program
- *Section*: A section for any course
- *Course*: Any class offered
- *Department*: A specific department at LionState university
- *Faculty*: Any supporting faculty member
- *Instructor*: Any faculty member who teaches a course
- *Assistant*: Any teaching assistant, learning assistant, grader for a course
- *Researcher*: Any faculty member involved in research
- *Research_Assistant*: Any research assistant not affiliated with the school
- *FacultyMember_Other*: Any other faculty member

The following ER diagram represents the relations of each of the aforementioned entities. In following sections, each relationship is described in detail.

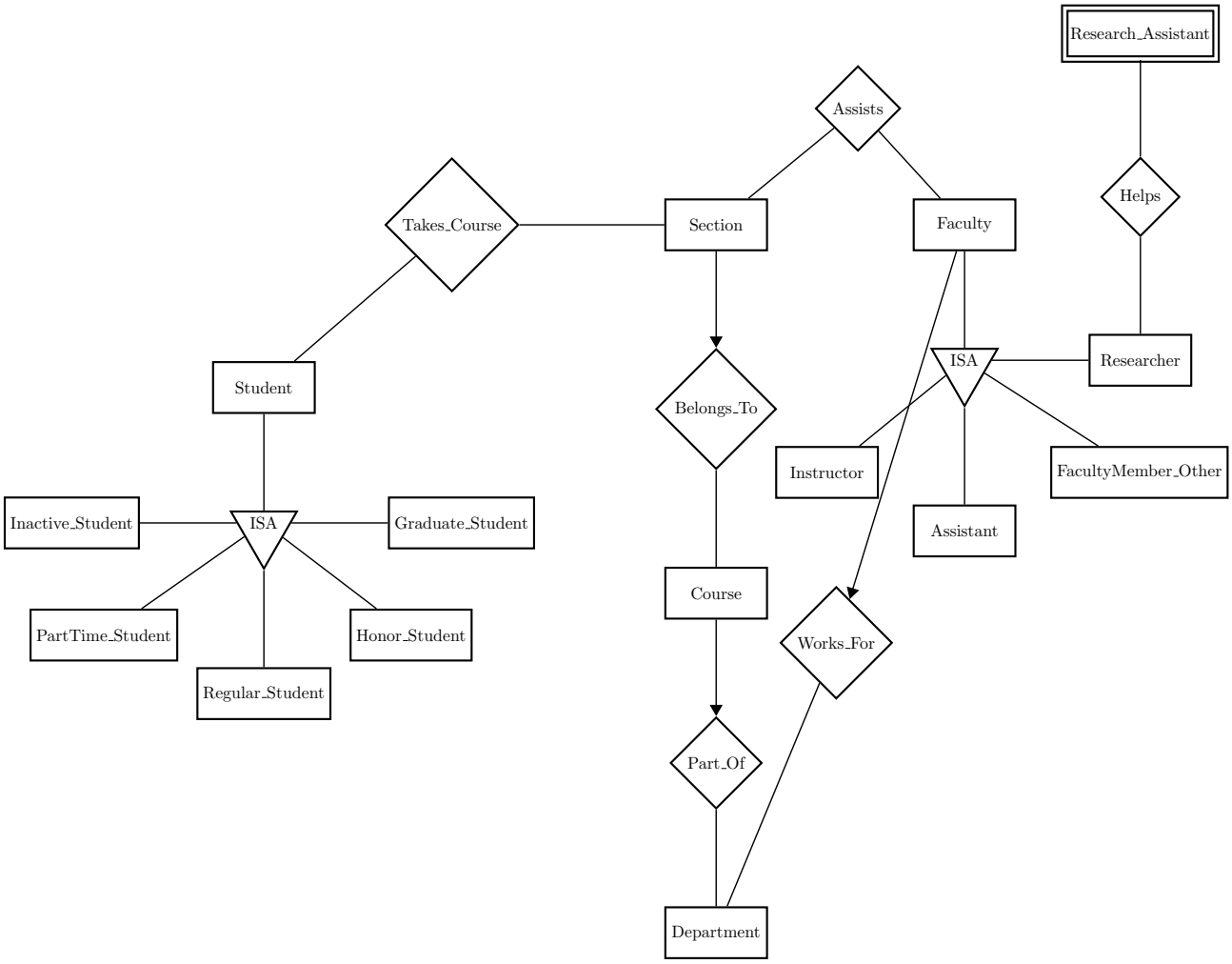

Figure 9: ER Diagram Overview

4.1 Student

A student is any user who is enrolled in LionState university. To break down a student further, we abstract the idea of a student into 5 categories:

- *Inactive_Student*: Any student who is inactive such as a past student or a student taking a break from school.
- *PartTime_Student*: Any student taking fewer than 12 credits.
- *Regular_Student*: Any regular student (Default)
- *Honor_Student*: Any student enrolled in the honor program
- *Graduate_Student*: Any student enrolled as in the graduate school.

The attributes of any student is listed below:

- Name: Student name

- ID: Student id
- Password: Password to log into system
- Age: age in years of student
- Email: Personal email address of student
- Gender: Male/Female/Other
- Street: Street Address of student
- City: City of student's permanenet address
- State: State of student's permanent address
- Major: Major of student

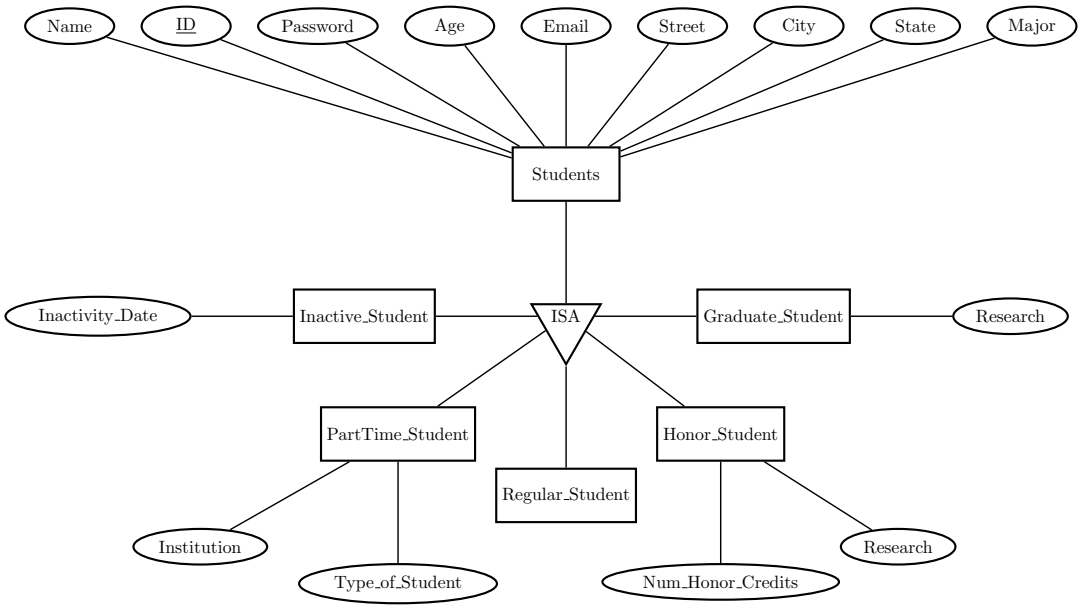

Figure 10: ER Diagram of student

4.1.1 Inactive Student

An inactive student represents any student who is enrolled at LionState University, but is currently taking no credits. This can include future students, past students, or students who simply are taking a break from school. The only difference that a an inactive student has over a regular student is the inactivity period. If the inactivity period remains too long, it is an indicator that the studne tcan be deleted from the database. The idea to abstract this student from a regular student comes from this exact reason: it is important to maintain this type of information easily. The only extended attribute is the inactivity date. The inactivity date will indicate an appropriate measurement for determining an appropriate date for termination.

- *Inactivity_Date*: The date of inactfivity of the student.

4.1.2 Part-Time Student

A part-time student is any student taking fewer than 12 credits during as semester. This ia an important distinction since it helps decide the student's tuition, available resources, and other aspects decided by the school's administration. Therefore, in order to reduce query time and computation time, a part-time student is derived from the student entity. A part-time student is assumed to be one of three options: (1) a student in a dual-enrollment program, (2) a student taking fewer credits, (3) a coop student. As a result, the type of student as well as the institution of the student, if available is stored.

- *Institution*: The intstitution of the student if in a dual-enrollment program (*NULL* if option 2 and *COOP* if option 3).
- *Type_Of_Student*: Type of student (1,2, or 3).

4.1.3 Regular Student

A regular student is the dfault student. Basically, any student who does not fall into one of the other categories is considered to be a regular student. For example, if a student takes more than 12 credits, is not enrolled in the honor school, and is an undergraduate student is a regular student if he/she is actively taking courses. As a result, no new attributes are needed.

4.1.4 Honor Student

An honor student is any student whom is enrolled in the honor program. Note that the honor program is only offered for the undergraduate program. As a result, no one student can be considered a graduate honor student for the same major. However, these are not necessarily mutually exclusive, however very likely. For example, a student can be a graduate student for computer science while being an honor student for music. A student cannot be a graduate student for computer science wile also being ahonor student for computer science. An honor student is also assumed to be active in some research activity, aw well as participating in a number of honor credits. An honor student is given more attribute, as a result:

- *Num_Hours_Complteted*: Number of honor credits botained so far.
- *Research*: Specific research program in which the student particpates.

4.1.5 Graduate Student

A graduate student is a special type of student who is in a completely different degree program as the rest of the students. Pu simply, they are any student attempting to obtain a master's, PHD, or other related degree after completion of an undergraduate degree. Usually, a graduate student must also particpate in some form of research like an honor studen mentioned in section 4.1.4. As a result, the same attribute is given to the graduate student.

- *Research*: Specific reasearch prgoram in which the student participates.

4.2 Courses

A course is group into three categories:

- Departments
- Course
- Section

After each level, the specific course get more and mroe detailed from the entire department type, all teh way to the specific section. For example. there are multiple computer science courses (Department is Computer Science). Within that department, there exists multiple coursess such as CMPSC 431W (Course is CMPSC 431W). However, within that specific course, tehre are multiple sections.

As a result, the entities are related in a linear fahion. The following diagram shows each in detail.

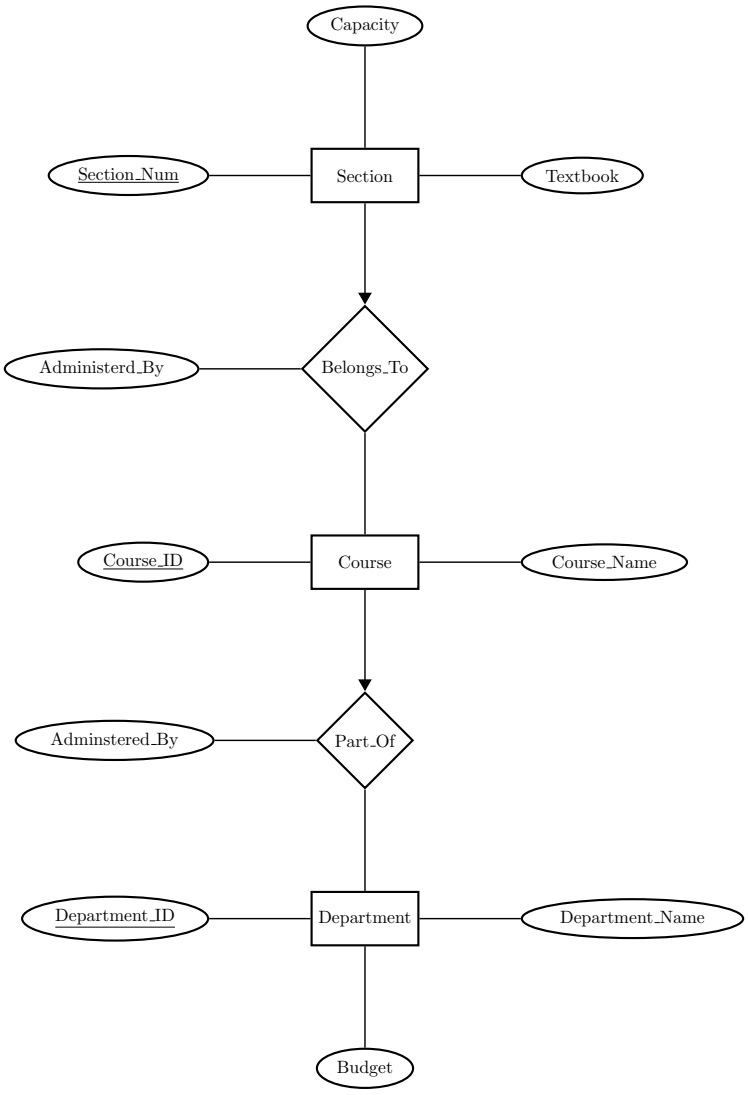

Figure 11: ER Diagram of Courses

4.2.1 Section

A section represents a single section of a course. Within the course, one can identify who is taking the course, who is teaching the course, student grades, and other aspects of the a class. However for the this specfic section, only the relationship betweem a secction, course, and department shall be mentioned. Withi this definithin, a course is given three aattributes:

- *Section_Num*: Identifier and section number of a section

- *Capactiy*: Capacity of the section
- *Textbook*: Specific textbook used for the course

4.2.2 Course

A course is a specific curricuulum that a group of section should follow. Doing so, a course represent s a class such as CMPSC 431W or more sepcifically 431W within the Computer Science department. A course is only given identiers to identify the course as everything else should be able to be determined through the section.

- *Course_ID*: Unique identifer for the course
- *Course_Name*: Name of the course.

4.2.3 Department

A department represents a group of courses tgat maintain a major or degree program. For example computer science, electirical engineering, chemistry and psychology are all valid departments. A department is more administrative than the rest of the groups so it has attributes regarding budget as well as its identiers.

- *Department_ID*: Unique idenfier for the department
- *Department_Name*: Name of the department
- *Budget*: Budger of the department

4.3 Faculty

A faculty member represents any person who assists in a class. Whether that be a teacher, teaching assistant, laboratory manager, etc.m that person is a faculy member. Because there are so many categories for a faculty members, a faculty member is abstracted in a way very similar to the student.

- Instructor: Instructor of a course
- Assistnat: Any teaching assistant, learning assistant, laboratory manager that supports the course
- Researcher: Any person participating in research affiliated with the school
- Other: Any other person who does not fit into the aforementioned categores such as an IT specialist or a janitor.

Note that these categroes are not necessarily mutually excliusive from each other. For instance, more often than not, a researcher will also teach a course, making them both a researcher and am instructor.

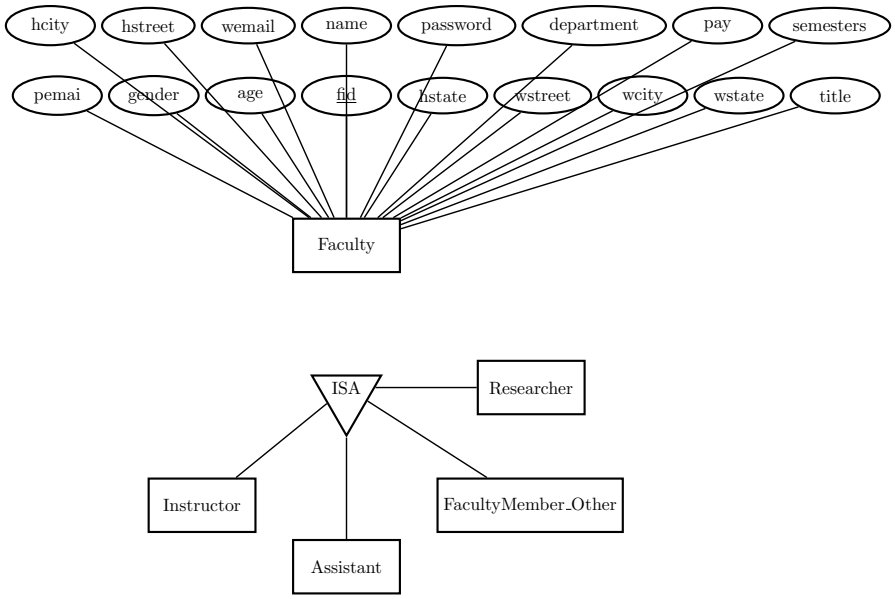

Figure 12: Overview ER diagram of Faculty members

4.4 Researcher

A researcher is any user who actively participates in research affiliated with LionState University. Note that we will not discard inactive researchers due to the lack of need and uncertainty of use. A researcher usually has a team behind him/her, so they must be accounted for as well. If that person is not affiliated with the University, however, issues may occur, so we use a weak entity relationship instead. The researchers have no additional attributes, so use section 4.3 as a reference.

5 Technological Survey

In this section, different options for implementing the database and web-application are mentioned and compared.

5.1 Intended Implementation

5.1.1 Angular JS

Angular JS is a framework, originally developed by Google in order to build very powerful Web applications. It is extremely useful for buling large scale web applications which need high performance, something that seems perfect for a database application. Further it is extremely easy to maintain and hundreds of web applications already used it, meaning it is robuts and reiliable. It utilizes JavaScript, a language that is extremely powerful for web applications, and is one of the best languages for such a task. Further, following a client-side Model-View-Controller pattern and Model-View-Viewmodel pattern, it aims to simplify the testing and development of web applications.

5.1.2 MYSQL Database

MySQL is an open-source relational database management system, derived from SQL. The program is written in C/C++ and is parsed by yacc in order to operate on multiple different platforms and cross-platform support. There exists stored procedures, triggers, cursors, information schea, and many other features that any database can take advantage.

5.2 Alternatives

5.2.1 Ruby on Rails

Ruby on Rails is a very useful, intuitive, and productive web application frameword written by David Hansson. It is very easy to use and one can create database-driven web application using a Model-View-Controller (MVC) pattern very easily. However, it is currently written in Ruby, so it is not the preffered method. However, many websites such as Github, AirBnB use this frameworks. If Angular JS is not suitable enoight for this application, Ruby on Rails is definitely the alternative.

5.2.2 Django

Django is one of the more robust framework intended for speed. It was invented in order to male applications fast and reilable. The framework is in Python, so there may be type checking errors, which is the primary reason for avoiding this framework. Otherwise, it is very strong. Because it is a high level framework, it takes care of a lot of the hassle usually required for web development. As it is also extremely scalable, it is suitable for database applications.

5.2.3 FLASK

Flask is a microframework for Python and is extremely easy to use. However, after previous attempts using this framework, there are samll quirks that are harder to deal with, which is why is should be avoided. However, it is an extremly strong framework that is easy to implement due to a lack of versatility. Since itdoes nore require any particular tools or libraries, it is classified as a microframework. However, it does not have any database application layer, making it unusable for this project.

6 Logical Database Design

This section will discuss the conceptual database design of the CanvasPath application.

6.1 Overview

The CanvasPath application will have primarily four tables or databases:

- Student
- Faculty Member
- Course
- Department
- Section
- ResearchAssistant

Each individual group is more specifically defined in section 4.

6.2 Student

Based on the student attributes mentioned in **??**, it is apprpriate to combinee all the abstracted student types and instead define a type attribute to conglomerate all the categories. The primary key will be the student id and the foreign key will be the section ID. Since sections are not deleted when a student is deleted, no action is taken on delete.

```
1       CREATE TABLE STUDENT
2         sid            INT NOT NULL,
3         type           INT NOT NULL,
4         name           VARCHAR[50] NOT NULL,
5         age            INT,
6         email          VARCHRA[50],
7         street         VARCHAR[50],
8         city           VARCHAR[50],
9         state          VARCHAR[2],
10        major          VARCHAR[40],
11        inactive_date  INT,
12        institution    VARCHAR[100],
13        type_student   VARCHAR[30],
14        honor_credit   INT,
15        GPA            REAL,
16        password       INT,
17        Research       VARCHAR[100]
18        sectid         INT,
19        PRIMARY KEY (sid),
20        FOREIGN KEY (sectid) REFERENCES FACULTY,
21        ON DELETE NO ACTION
22        );
```

6.3 Faculty Member

Based on the faculty attributes mentioned in **??**, it is also appropriate to combine all the abstracted faculty types into one table. However the researcher serves as a special case because a weak entity relies on it. Alike thes student if a faculty member is removed, a course is not removed.

```
CREATE TABLE FACULTY(
  fid           INT NOT NULL,
  type          INT NOT NULL,
  name          VARCHAR[50] NOT NULL,
  age           INT,
  pemail        VARCHRA[50],
  pstreet       VARCHAR[50],
  pcity         VARCHAR[50],
  pstate        VARCHAR[2],
  wemail        VARCHRA[50],
  wstreet       VARCHAR[50],
  wcity         VARCHAR[50],
  wstate        VARCHAR[2],
  dept          VARCHAR[40],
  rating        REAL,
  password      INT,
  payment       VARCHAR[100]
  sectid        INT,
  PRIMARY KEY (fid),
  FOREIGN KEY (sectid) REFERENCES STUDENT,
  ON DELETE NO ACTION
  );
```

6.4 Section, Courses, and Departments

The section represent courses and should store everything related to that course. Following the attributes in 4.2.1, the section should connect students with faculty members. The primary key is the course ID and foreign keys are both the faculty id ad student id.

```
CREATE TABLE SECTION(
  cid           INT NOT NULL,
  sid           INT NOT NULL,
  fid           INT NOT NULL,
  capactiy      INT,
  sec_name      INT NOT NULL,
  textbook      VARCHAR[200],
  administrator VARCHAR[200],
  course_id     INT NOT NULL,
  PRIMARY KEY (cid),
  FOREIGN KEY (sid) REFERENCE STUDENT,
  FOREIGN KEY (cid) REFERENCE FACULTY,
  FOREIGN KEY (course_id) REFERENCE COURSE
  ON DELETE NO ACTION
  );
```

The same can be done for the course:

```
CREATE TABLE COURSE(
  course_id      INT NOT NULL,
  course_name    VARCHAR[50] NOT NULL,
  administrator  VARCHAR[200],
  dept_id        INT NOT NULL,
  PRIMARY KEY (course_id),
  FOREIGN KEY (dept_id) REFERENCE DEPARTMENT,
  ON DELETE NO ACTION
  );
```

And finally for the department:

```
CREATE TABLE DEPARTMENT(
  dept_id        INT NOT NULL,
  dept_name      VARCHAR[50] NOT NULL,
  administrator  VARCHAR[200],
  budgets        REAL,
  PRIMARY KEY (dept_id),
  ON DELETE CASCADE
  );
```

6.5 Reaearch Assistant

Since the research assistant is a weak entity, the following table must be issued:

```
  CREATE TABLE RESEARCH_ASSISTANT(
    rid      INT NOT NULL,
    fid    INT NOT NULL,
    dept_id      INT NOT NULL,
    PRIMARY KEY (rid_id),
    FOREIGN KEY (fid) REFERENCE FACULTY,
    ON DELETE NO ACTION
    );
```

7 Appendix

References

[1] Ramakrishnan; Gehrke. *Database Management Systems- Third Edition*. McGraw-Hill Higher Edication, 2003.

[2] Ronacher, Armin. *Flask docs Forward*. http://flask.pocoo.org:80/docs/0.10/foreword 2013.

[3] Google. *What is AngualarJS*. https://docs.angularjs.org/guide/introduction 2018.

[4] Oracle. *MySQL 8.0 Reference Manual*. https://dev.mysql.com/doc/refman/8.0/en/ 2019.

[5] Bigg, Ryan. *Getting Started with Rails*. https://guides.rubyonrails.org/getting_started.html 2018.

[6] Django. *Why Django?* https://www.djangoproject.com/start/overview/ 2019.