Student Employee Scheduling Management system

SU Food Services

Created by:
Avin Deshmukh
Mahesh Kumar
Lavnish Talreja
IST 659 M006 Fall 2019,
School of Information Studies,
Syracuse University

Contents:

Overview:

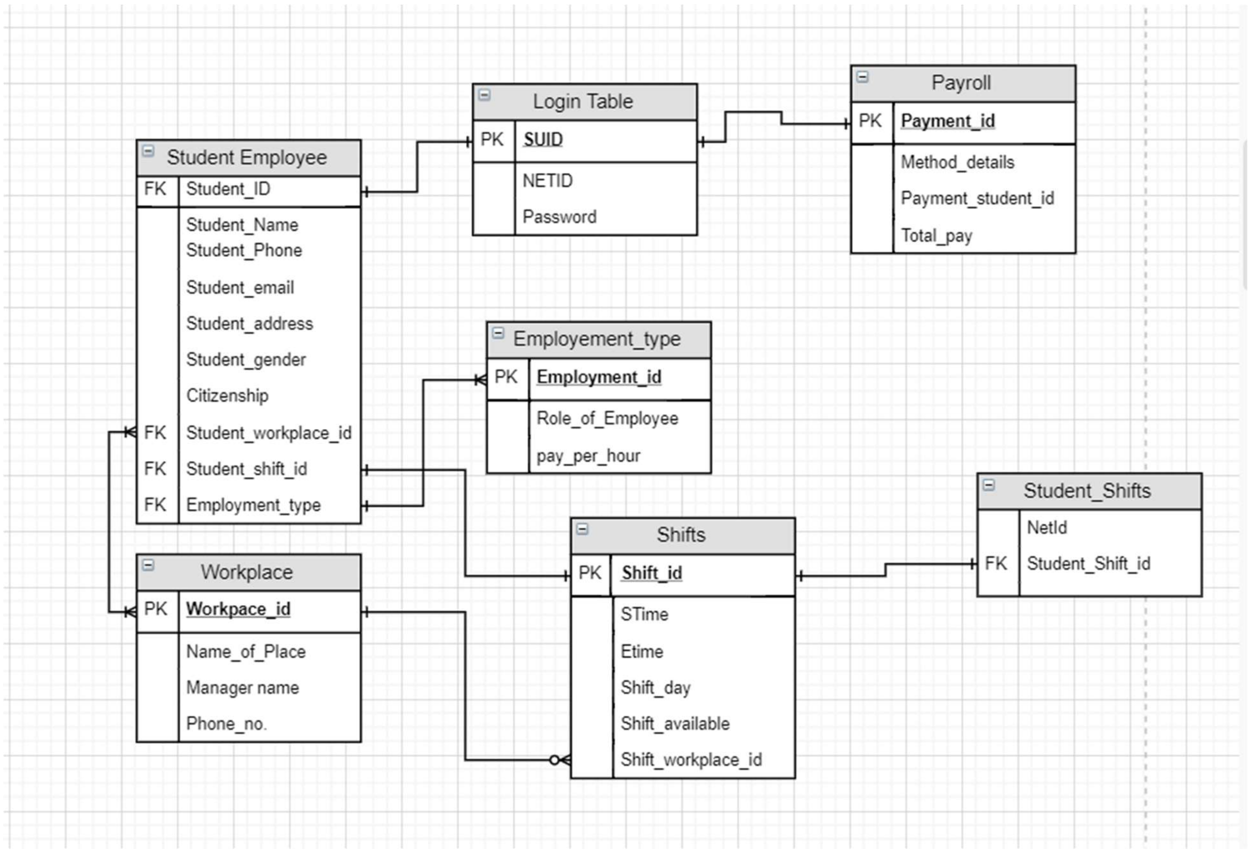
Automation of employee scheduling is one of the most valuable productivity improvements an organization can make to save a considerable amount of time. The database management project implemented is for a scheduling system designed for Syracuse Food services. This project mainly focuses on the data generated inside a food service and proposes a solution to manage it effectively and in a secure manner. The system would be able to keep student employee records and would allow them to select a temporary shift whenever available so that they can work few extra hours. Moreover, in addition to student employee personal details like name, phone, address, email etc , it also consists of workplace details such as food cafes/dining halls a student wants to work with, the temporary shifts available, their start and end timings and whether those shifts are available or not.

An employee can also access their payrolls, also the system ensures that if the student has a citizenship apart from US, that student cannot take temporary shifts more than 20 hours. A student can also change the mode of payment as well. Moreover, a student can even drop a shift if he/she does not want to take it anymore. And the database would be updated with the actions taken by a student accordingly. Thus, it digitizes the manual selection of temporary shifts for dinings and cafes. Thus, this project eliminates the human error that was caused during student employee scheduling. The data can be easily accessed, easily monitored and a new schedule can be easily created.

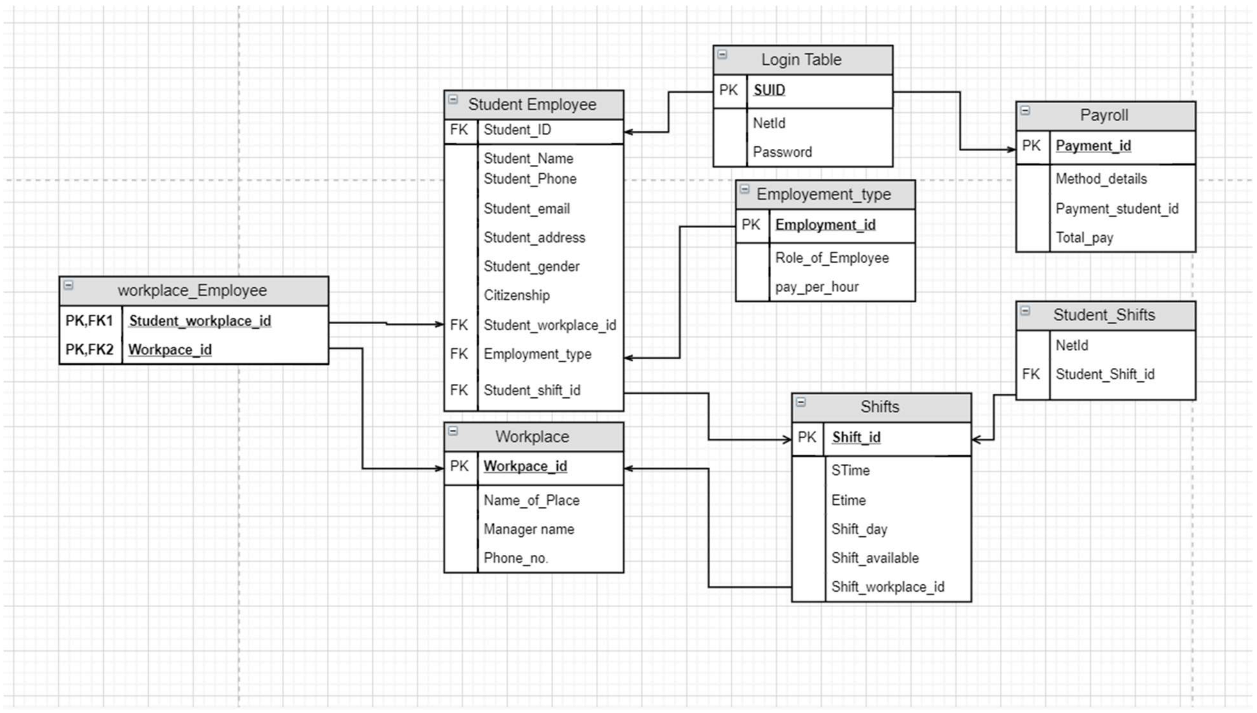
Business rules:

1. A student employee can work for more than one workplace (café's/dining halls).
2. A café/dining hall can have one or more than one student.
3. A student can have only one Login username and Password.
4. A unique combination of Login username and password can be accessed by only one student.
5. A student can have access to only one payroll system.
6. A payroll system can be accessed by a unique login of student employee.
7. A student can take multiple shifts at a workplace.
8. A shift can have one or more student employees.

Conceptual Data Model Diagram:



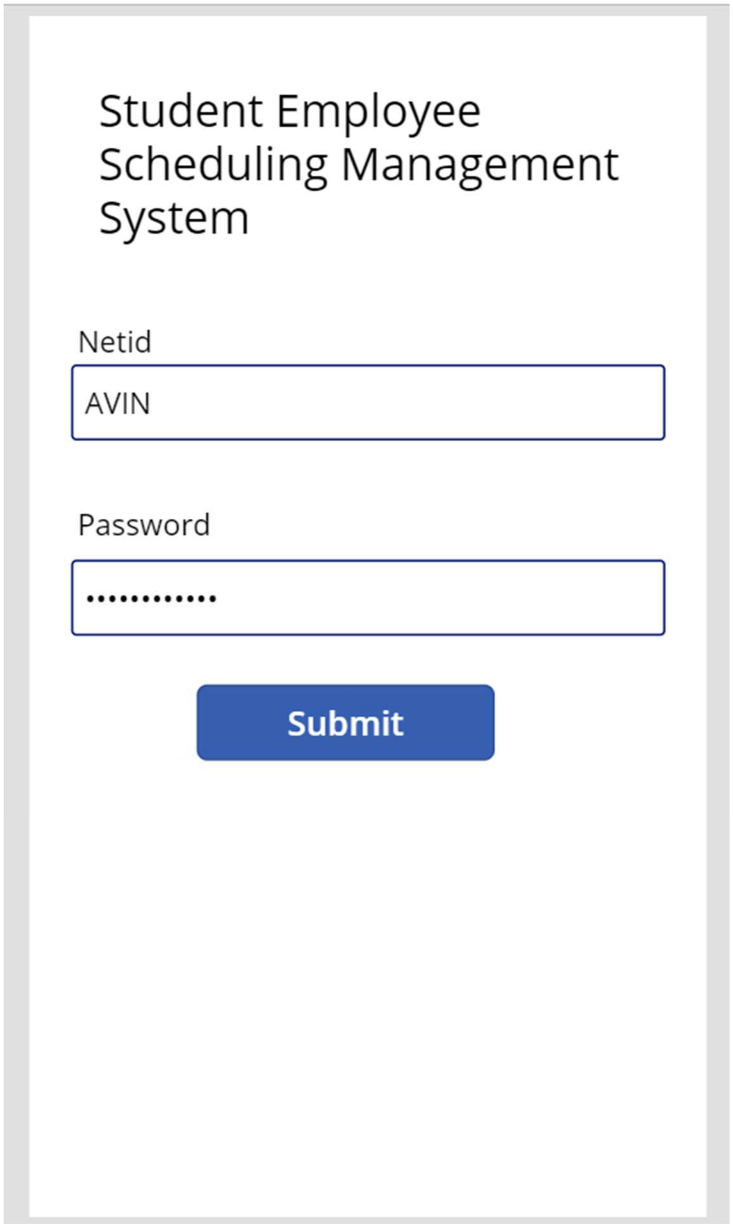
Logical Data Model Diagram:

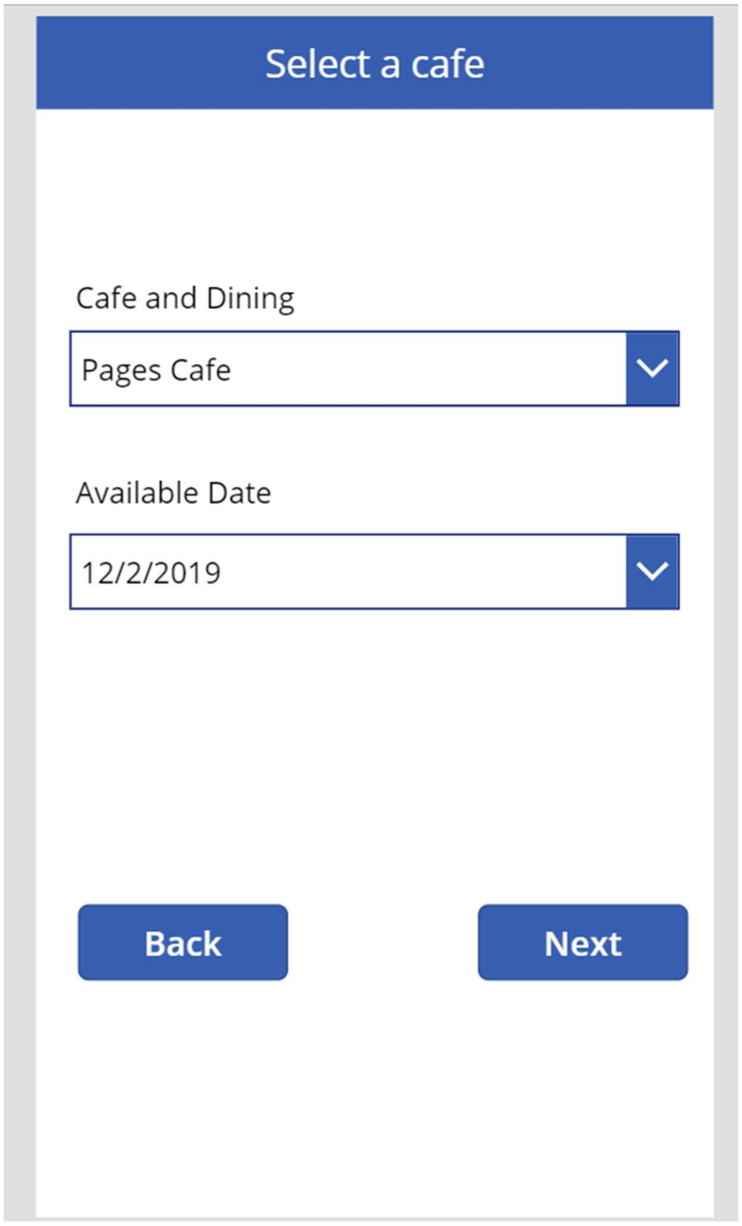


SQL CODE:

```
-- created Table for student employees
drop table admin;

create table admin
(
SUID numeric primary key,
password varchar(20) not null
)

select *from admin;

drop table student_employees;
create table student_employees
(
   Student_id numeric not null foreign key references admin(SUID),
   student_name varchar(50) not null,
   student_phone int not null,
   student_address varchar(100) not null,
   student_gender char not null,
   student_email varchar(20) unique not null,
   citizenship Varchar(20) not null,
   student_workplace_id numeric not null foreign key references workplace(workplace_id),
   student_employment_id varchar not null foreign key references
Employment_type(employment_id)
   )

  select * from student_employees;

  drop table Employment_type
  Create table Employment_type
  (
  employment_id varchar Primary key,
  role_of_employee varchar(50) not null ,
  pay_per_hour numeric not null
  )
  select * from Employment_type

  drop table workplace;
   create table Workplace
(
Workplace_id numeric Primary key,
Name_of_Place Varchar(100) not null,
Phone_no int not null,
Manager_Name Varchar(100) not null,
)

select * from Workplace;

drop table shifts;
create table shifts
(Shift_id numeric primary key,
Stime time not null,
Etime time not null,
Shift_day varchar(20) not null,
shift_workplace_id numeric not null foreign key references workplace(workplace_id),
shift_availability char not null
)

select * from shifts;

drop table payroll;
create table payroll
(payment_id numeric primary key,
method_details varchar(50) not null,
payment_student_id numeric not null foreign key references admin(SUID),
total_pay numeric not null
)
select * from payroll;

---Student shifts
drop table student_shifts;
create table student_shifts
(
netid varchar,
student_shift_id numeric not null foreign key references shifts(Shift_id)
);


--Stored procedure for login
drop procedure login_proc
create procedure login_proc
@Username varchar(20),
@Password varchar(20),
@Role varchar(25) OUTPUT
AS
   SET NOCOUNT ON
BEGIN

    If Not Exists (Select 1 From admin Where netid = @UserName) Set @Role = 'Incorrect
UserName'
    Else If Not Exists (Select 1 From admin Where passwords = @Password) Set @Role =
'Incorrect Password'
    Else Set @Role = 'Logged in Successfully'

    Select @Role

       END
       select * from shifts


       --procedure for employee_limit
       drop procedure employee_limit
       create procedure employee_limit
       @count int output
       as
       SET NOCOUNT ON
BEGIN
            declare @limit int='10'
       begin try
       set @count=(select sum(DATEDIFF(HOUR,stime,etime)) as total_time from shifts join
Workplace
       on  shifts.shift_workplace_id=Workplace.Workplace_id join student_employees
       on shifts.shift_id=student_employees.student_shift_id where shift_availability='Y'
       group by student_employees.Student_id) ;
       end try
       begin catch

       If @count<@limit
       update student_employees
       set student_shift_id=shifts.Shift_id from shifts
       where student_employees.student_shift_id=shifts.Shift_id
       else
       ROLLBACK TRANSACTION
        SELECT ERROR_NUMBER(), ERROR_MESSAGE();
   END CATCH
       end

       --function to calculate the payroll of an employee
       drop function total_pay_func
       create function total_pay_func
       (@ret numeric)
       returns numeric
AS
-- Returns the total pay of a student employee
BEGIN
    SELECT @ret = e.pay_per_hour *p.total_pay
       from Employment_type e join payroll p on
       e.employment_id=p.payment_student_id;
     IF (@ret IS NULL)
        SET @ret = 0;
    RETURN @ret;
END
```

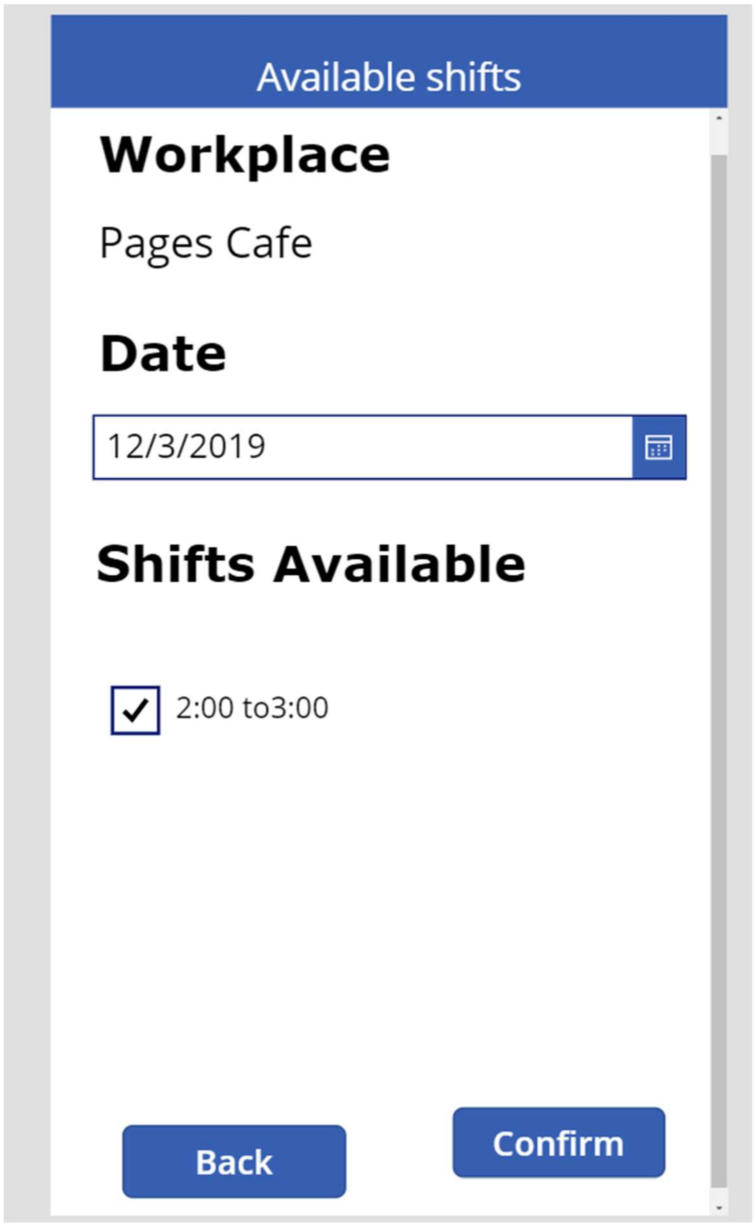
Diagrams of Screen used in the application:





Cafe and Dining: Pages Cafe

Available Date: 12/2/2019



Available shifts

Workplace

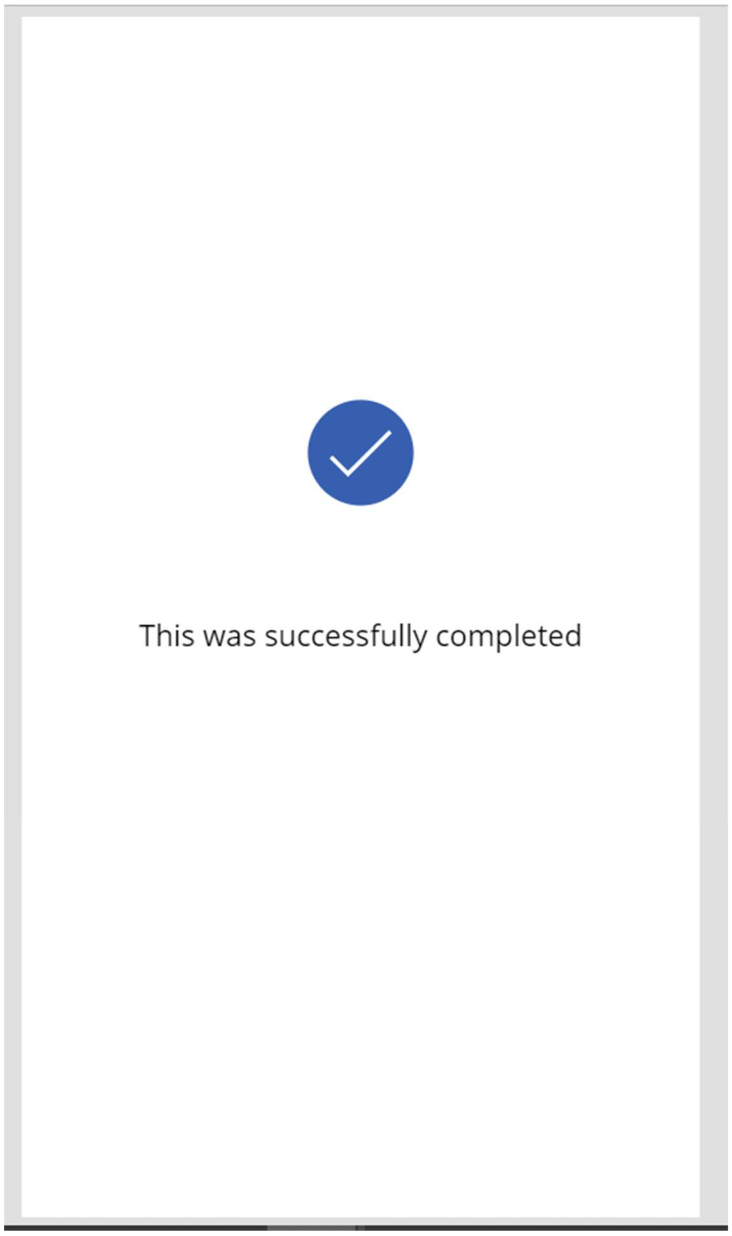
Pages Cafe

Date

12/3/2019

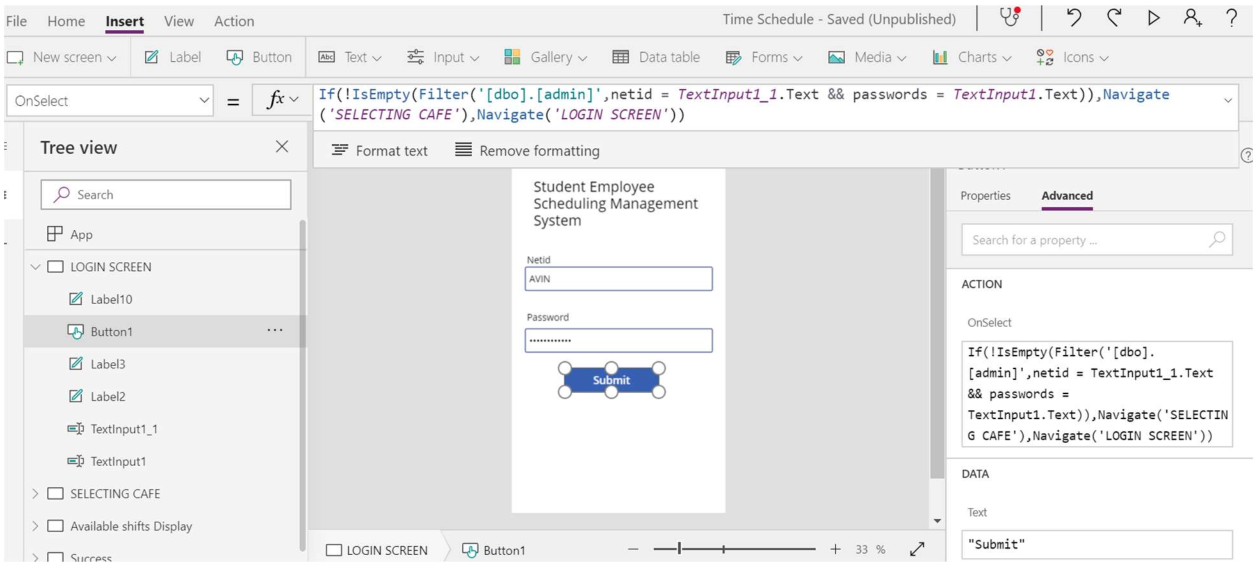
Shifts Available

[✓] 2:00 to3:00



This was successfully completed

Implementation in Power Apps:

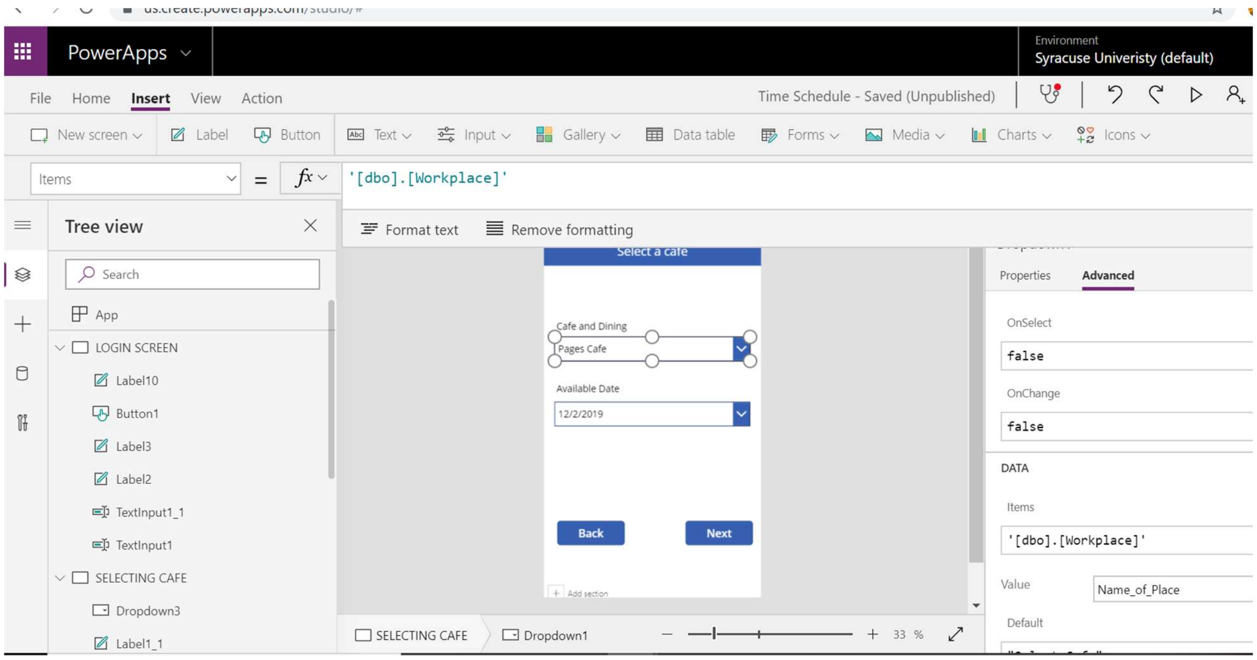


File Home Insert View Action
Time Schedule - Saved (Unpublished)
New screen Label Button Text Input Gallery Data table Forms Media Charts Icons
Tree view
Format text Remove formatting
App
LOGIN SCREEN
Label10
Button1
Label3
Label2
TextInput1_1
TextInput1
SELECTING CAFE
Available shifts Display
Success
Student Employee Scheduling Management System
Netid
Password
Submit
Properties Advanced
ACTION
DATA

OnSelect

```
If(!IsEmpty(Filter('[dbo].[admin]',netid = TextInput1_1.Text && passwords = TextInput1.Text)),Navigate
('SELECTING CAFE'),Navigate('LOGIN SCREEN'))
```

Text: "Submit"



Items

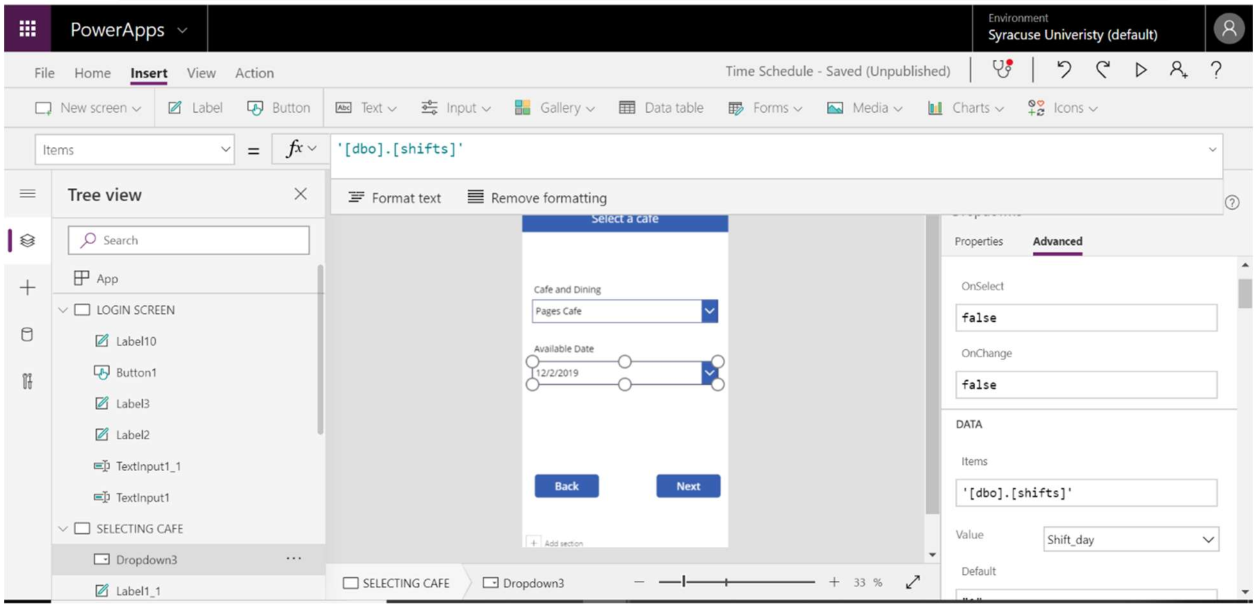
```
'[dbo].[Workplace]'
```

OnSelect: false

OnChange: false

Items: '[dbo].[Workplace]'

Value: Name_of_Place



Items = `'[dbo].[shifts]'`

OnSelect: `false`

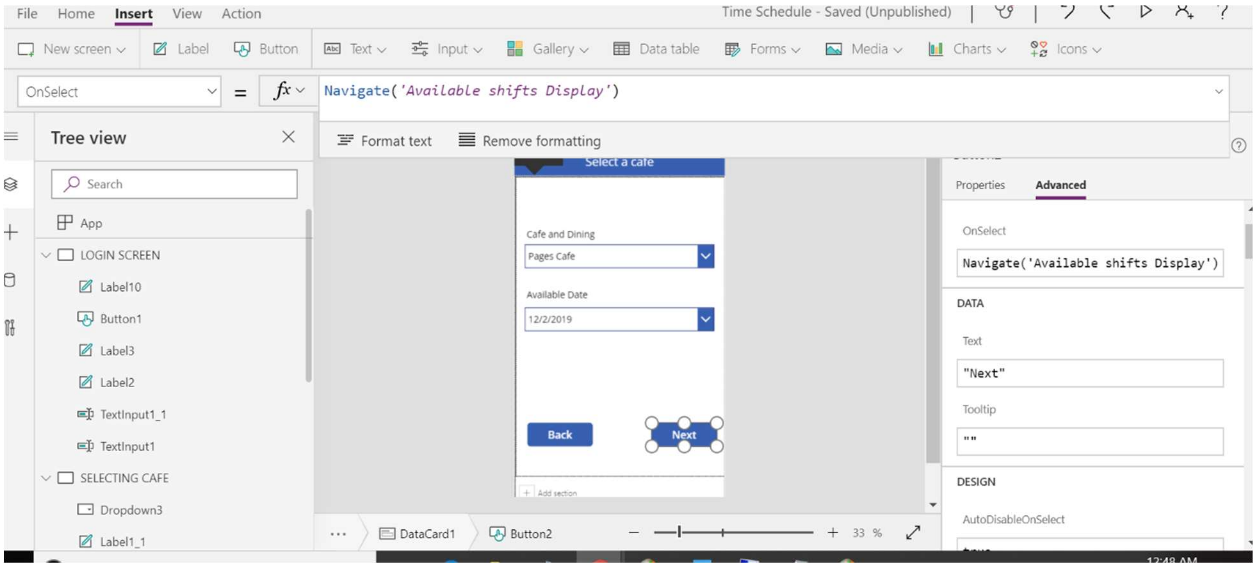
OnChange: `false`

DATA

Items: `'[dbo].[shifts]'`

Value: Shift_day

Default



OnSelect = `Navigate('Available shifts Display')`

OnSelect: `Navigate('Available shifts Display')`

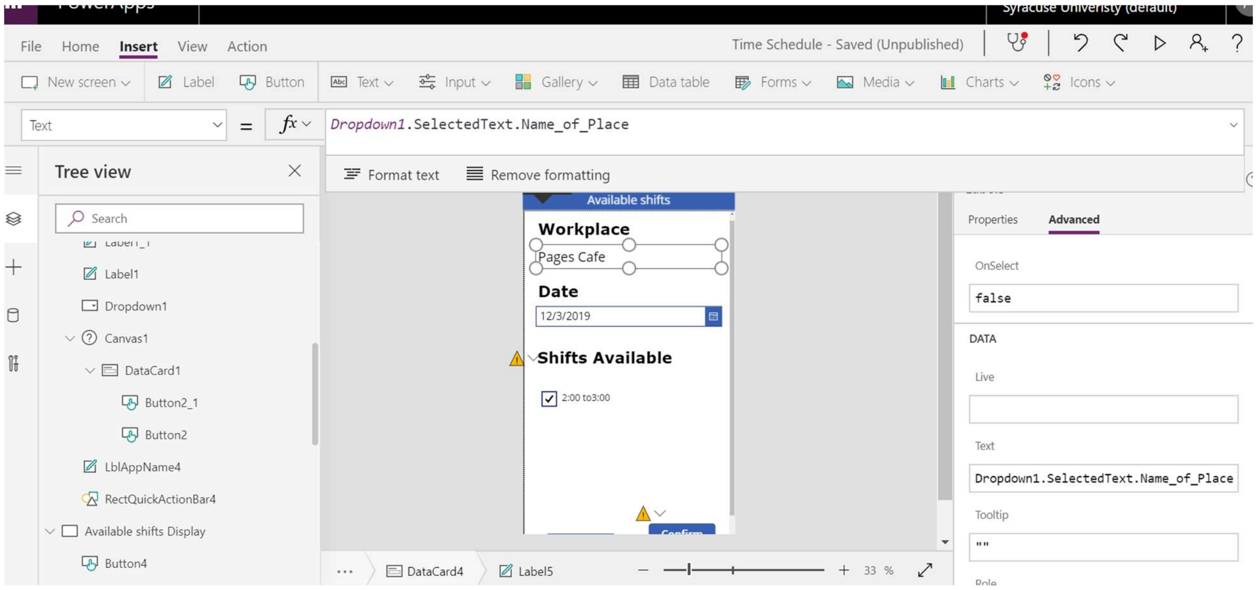
DATA

Text: `"Next"`

Tooltip: `""`

DESIGN

AutoDisableOnSelect



Text = Dropdown1.SelectedText.Name_of_Place

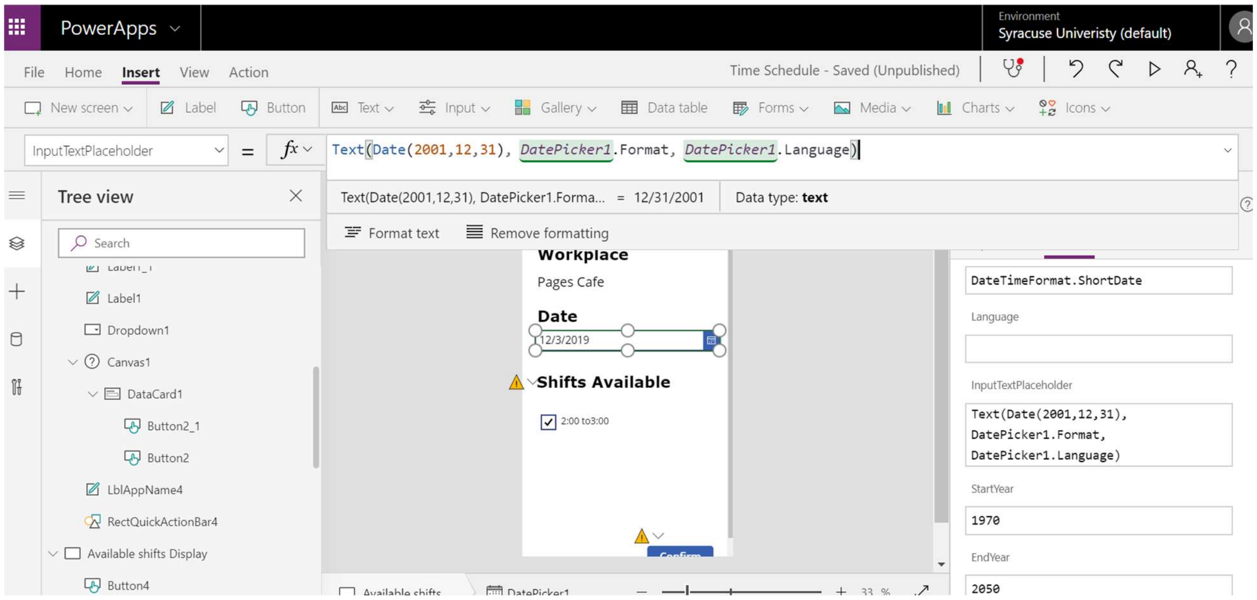
OnSelect: false

DATA

Live:

Text: Dropdown1.SelectedText.Name_of_Place

Tooltip: ""



InputTextPlaceholder = Text(Date(2001,12,31), DatePicker1.Format, DatePicker1.Language)

Text(Date(2001,12,31), DatePicker1.Forma... = 12/31/2001 Data type: **text**

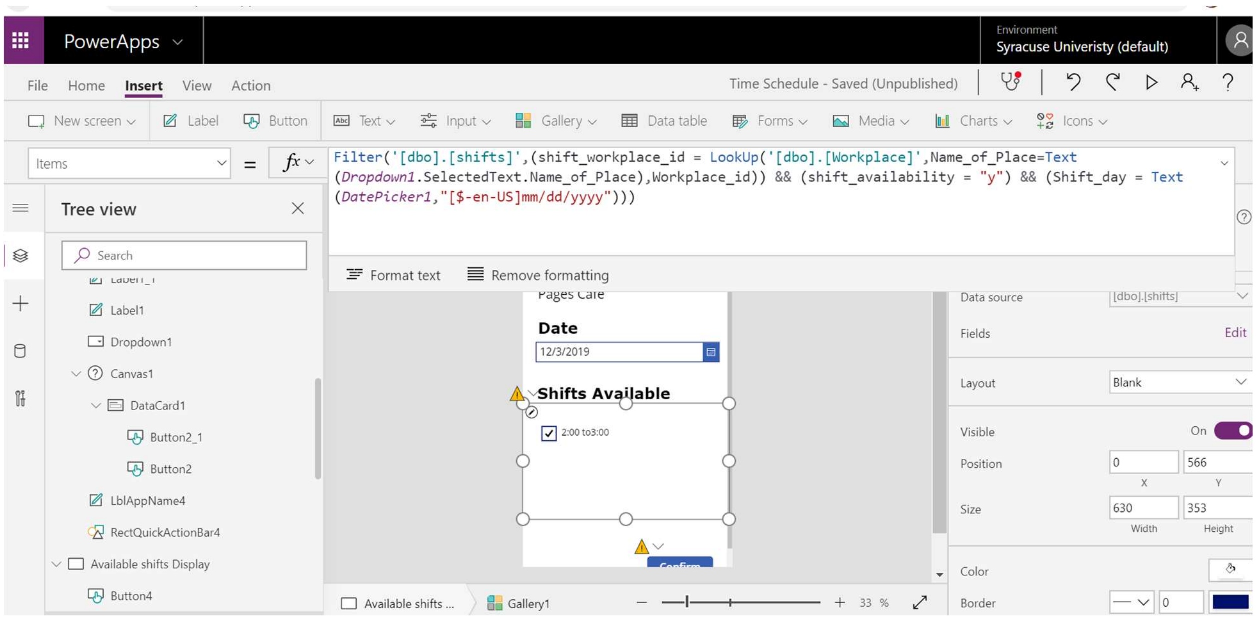
DateTimeFormat.ShortDate

Language:

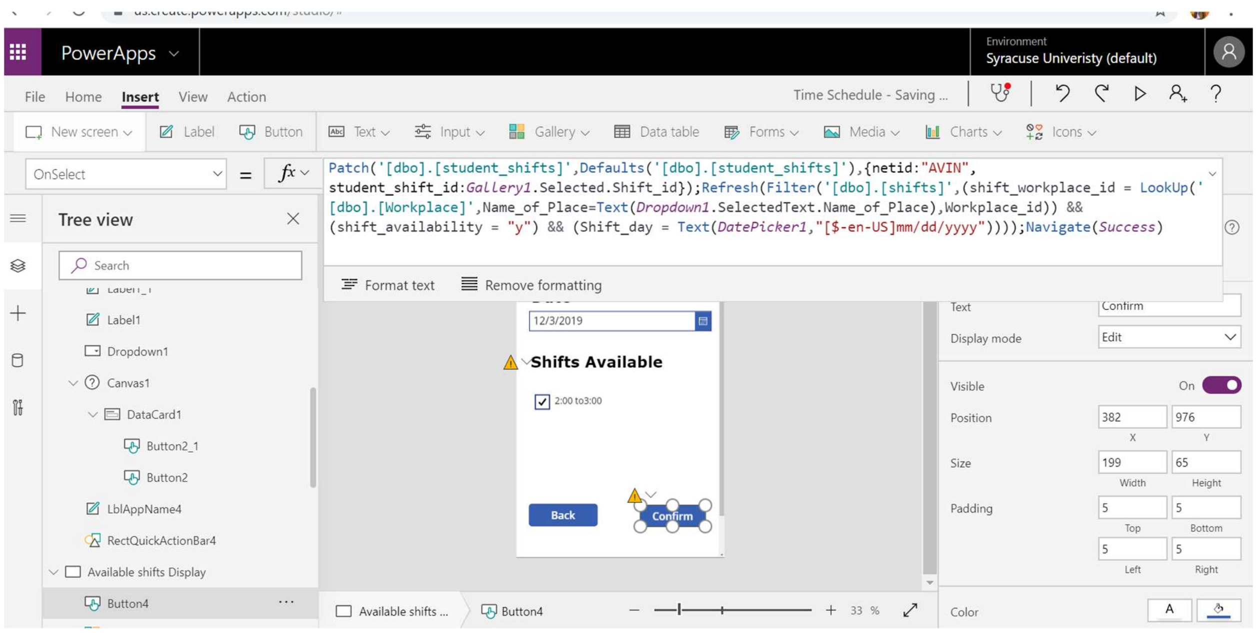
InputTextPlaceholder: Text(Date(2001,12,31), DatePicker1.Format, DatePicker1.Language)

StartYear: 1970

EndYear: 2050


Items

```
Filter('[dbo].[shifts]',(shift_workplace_id = LookUp('[dbo].[Workplace]',Name_of_Place=Text
(Dropdown1.SelectedText.Name_of_Place),Workplace_id)) && (shift_availability = "y") && (Shift_day = Text
(DatePicker1,"[$-en-US]mm/dd/yyyy")))
```


OnSelect

```
Patch('[dbo].[student_shifts]',Defaults('[dbo].[student_shifts]'),{netid:"AVIN",
student_shift_id:Gallery1.Selected.Shift_id});Refresh(Filter('[dbo].[shifts]',(shift_workplace_id = LookUp('
[dbo].[Workplace]',Name_of_Place=Text(Dropdown1.SelectedText.Name_of_Place),Workplace_id)) &&
(shift_availability = "y") && (Shift_day = Text(DatePicker1,"[$-en-US]mm/dd/yyyy"))));Navigate(Success)
```

LOG REPORT:

21st November:

Team:

1.Discussed the possibilities of the task and requirements of the end user.

2. Finalized the tables and attributes to be considered.

23rd November:

Team:

1)Created ERD on draw.io

2)Brainstorming on the ERD for the possible missing features.

25th November:

Avin - created tables

Mahesh- inserted data values in tables

Lavnish - added SQL constraints

27th November:

Avin - Rechecked the queries from previous meeting.

Mahesh- Added triggers and views

Lavnish - added stored procedures and functions

28th November:

Mahesh - Started Front End on Power Apps.

Avin- Created Login page, student employee page.

Lavnish - connected database with Power Apps

29 November

Team went for shopping together to spend some quality time:p

30 November:

Avin- created table of contents

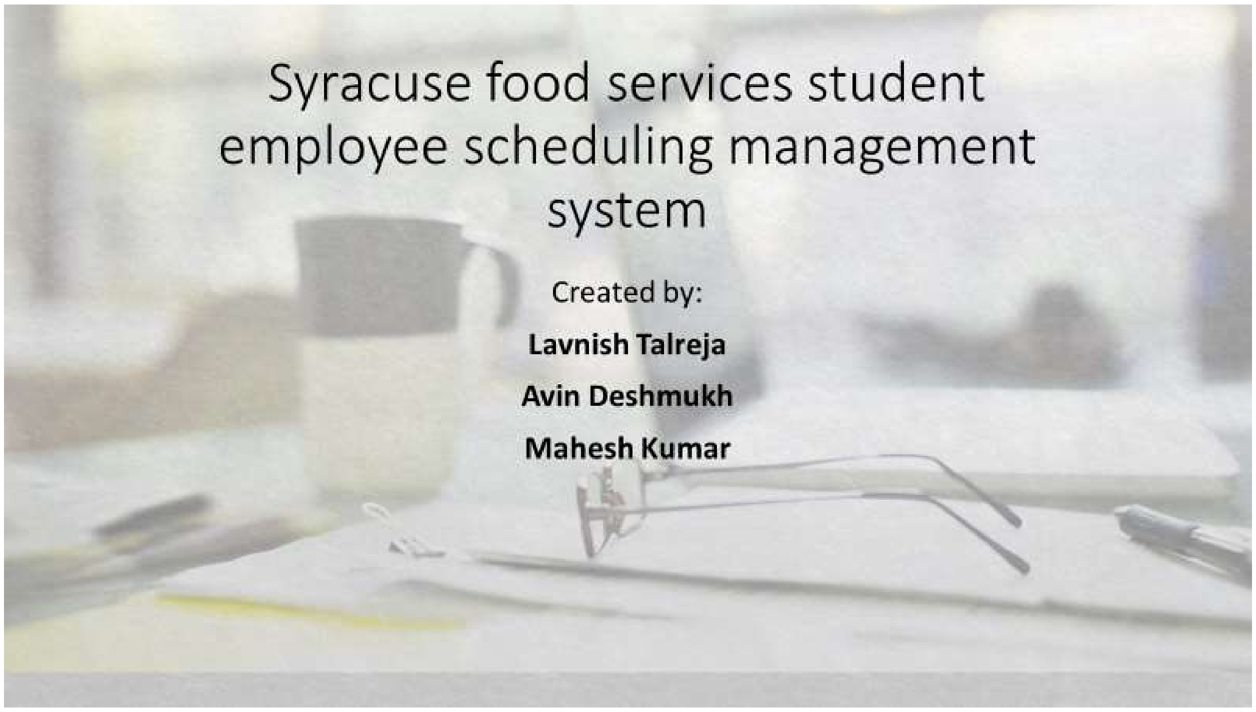
Mahesh- did the quality testing the demo app

Lavnish - Made the presentation for the app

1 December:

Team recorded a video where we explained the presentation and showed a demo of our app.

Presentation:



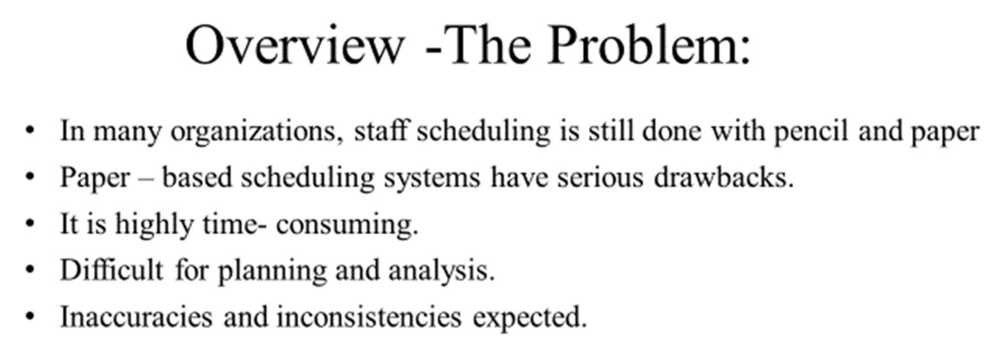
Syracuse food services student employee scheduling management system

Created by:

Lavnish Talreja

Avin Deshmukh

Mahesh Kumar



Overview -The Problem:

- In many organizations, staff scheduling is still done with pencil and paper
- Paper – based scheduling systems have serious drawbacks.
- It is highly time- consuming.
- Difficult for planning and analysis.
- Inaccuracies and inconsistencies expected.

Solution:

- Designing an employee system which automates the scheduling part for the system.
- Employees can be assigned shifts as per the availability.
- Employees can choose a temporary shift based on pay provided by a workplace and select a shift accordingly.
- An employee can even drop a shift.

Benefits:

- Automated the process of manually inserting employee details during shift selection.
- Eliminated human error caused during employee scheduling.
- Data can be accessed whenever there is a need to create a new schedule.
- Final product can provide a rich scheduling database for effective planning, analysis and budgeting.

Challenges Faced:

- Implementing each screen in power apps.
- Adding conditions, inserting and updating into tables using power apps.
- Analyzing functional requirements for the designing of database.
- Implementation of constraints such as Work limit for an employee, login system in powerapps.

If more time was permitted:

- Priority scheduling system could have been implemented.
- Implementation of Logoff system.
- Actions taken by employee during shift selection could have been monitored by an administrator.

Demo Video:

Part1:

https://drive.google.com/file/d/1VvC1oXQyaDbGs6gUDeoQRPAgoc7HNunI/view?ts=5de5fda8

Part2:

https://drive.google.com/file/d/1VzYj-FbNeWB0CvrIA8AIrK1jzmqK5gFH/view?ts=5de5fe03

Presentation Video:

https://drive.google.com/file/d/1BuofSncxeYQ-6i98Aoxvi4iUajxIffb5/view