**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 and Business Requirements…………………………………………3**

**Conceptual Data Model Diagram…………………………………………………4**

**Logical Data Model Diagram……………………………………………………….5**

**SQL Implementation code…………………………………………………………..6**

**Diagrams used in Power Apps……………………………………………………..9**

**Implementation in Power Apps…………………………………………………..13**

**Log Report…………………………………………………………………………………17**

**Team Presentation…………………………………………………………………….19**

**Video………………………………………………………………………………………….22**

**Overview:**

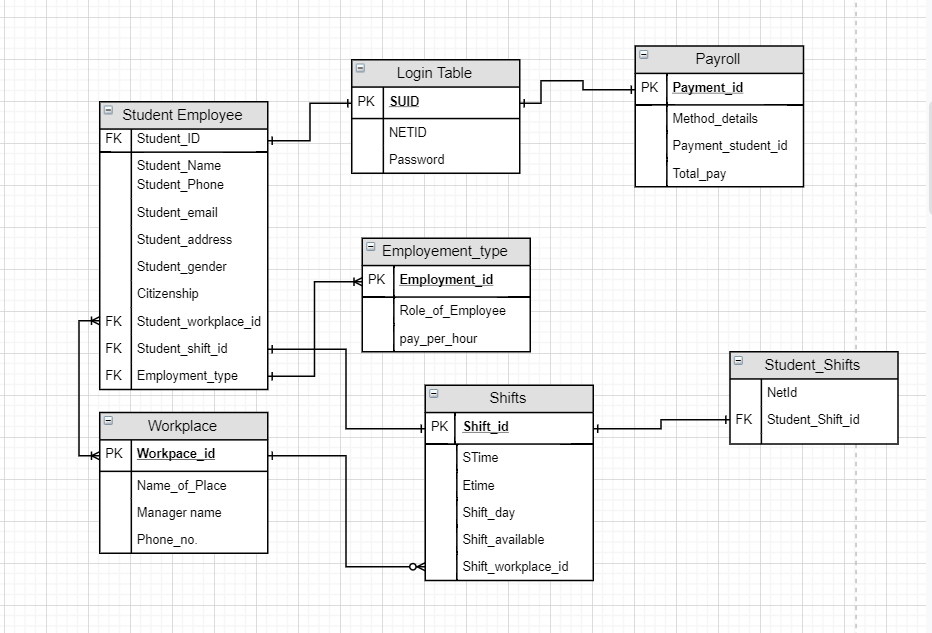
Automation of employee scheduling is one of the most valuable productivity improvements an  
organization can make to save a considerable amount of time. The database management project implemented is for a scheduling system designed for Syracuse Food services. This project mainly focuses on the data 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.

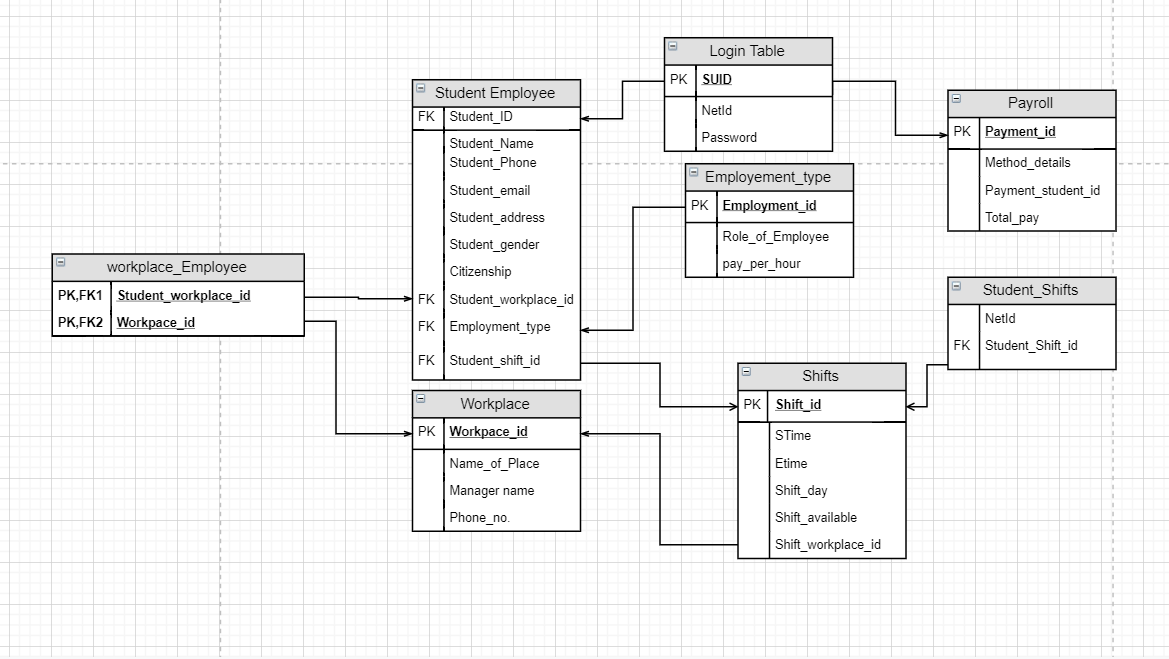
**Business rules:**

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;

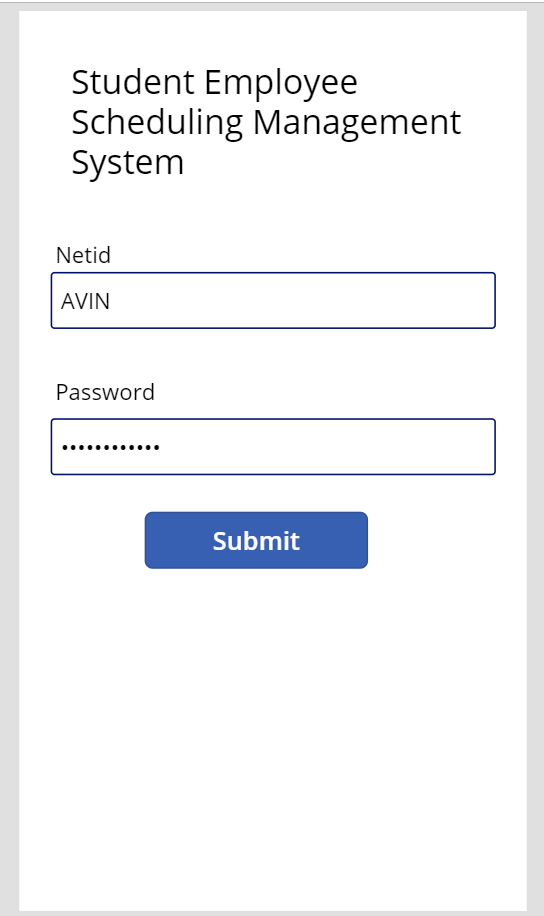
IF (@ret IS NULL)

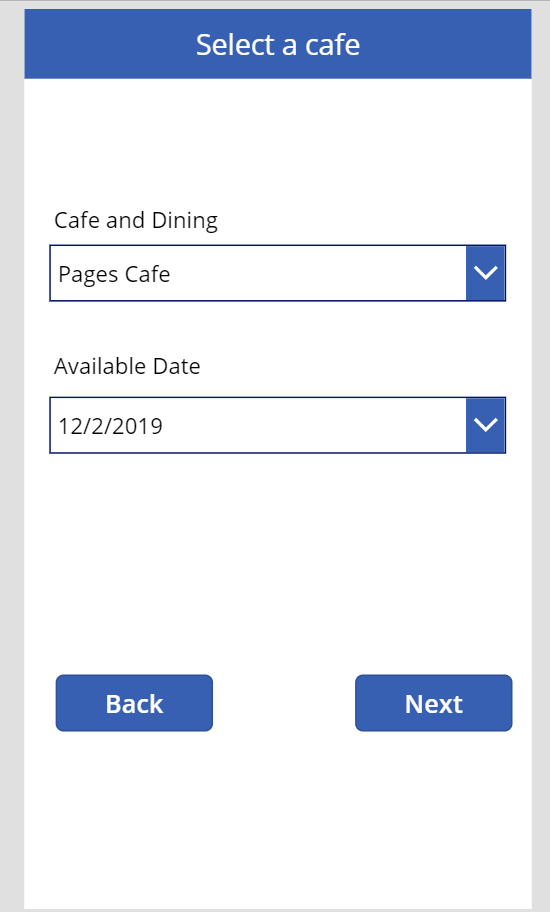
SET @ret = 0;

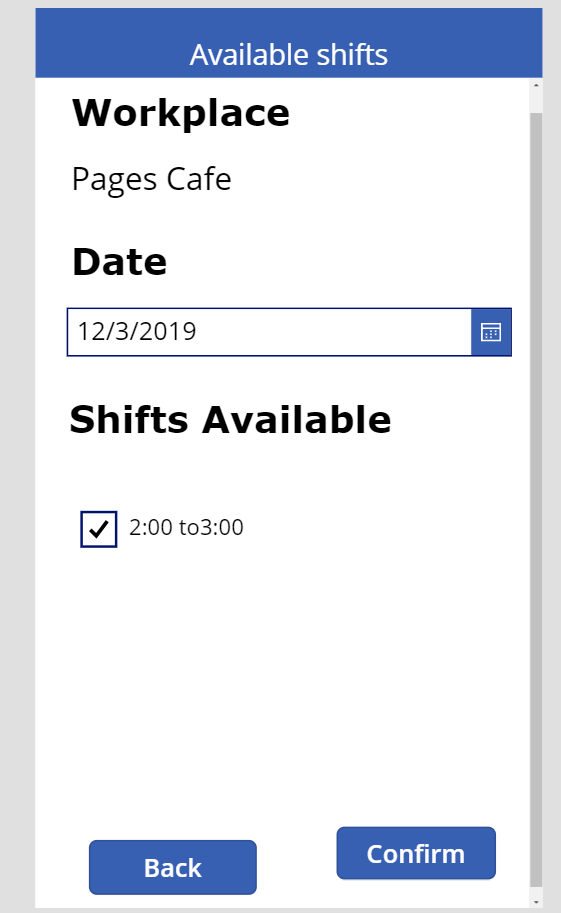
RETURN @ret;

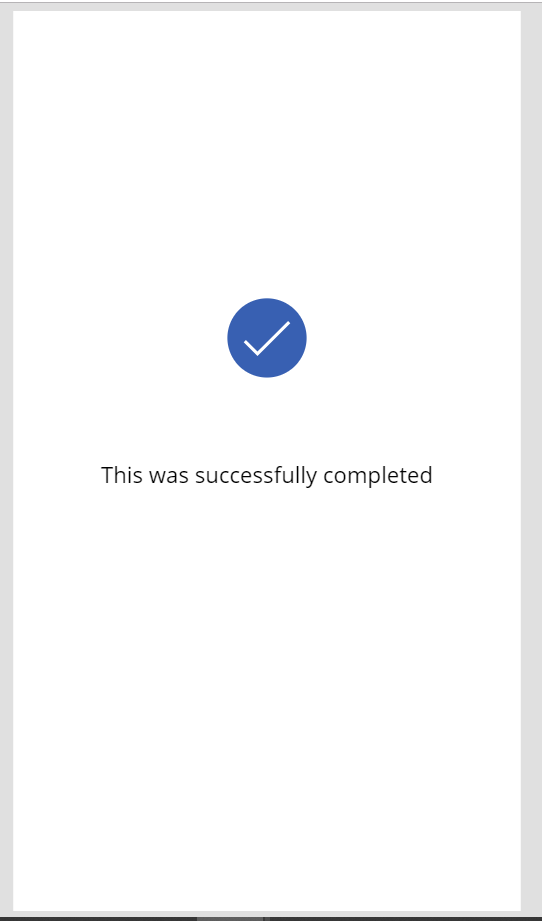
END

**Diagrams of Screen used in the application:**

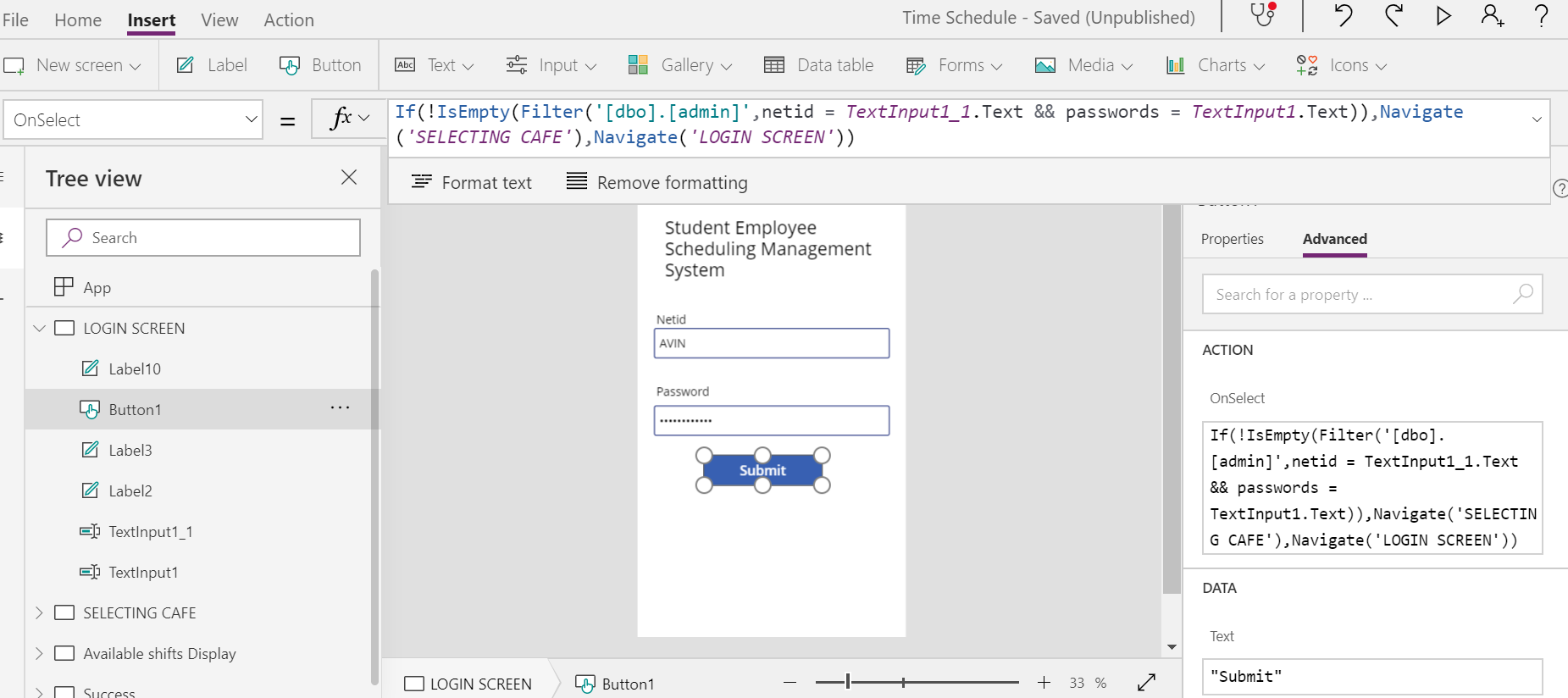


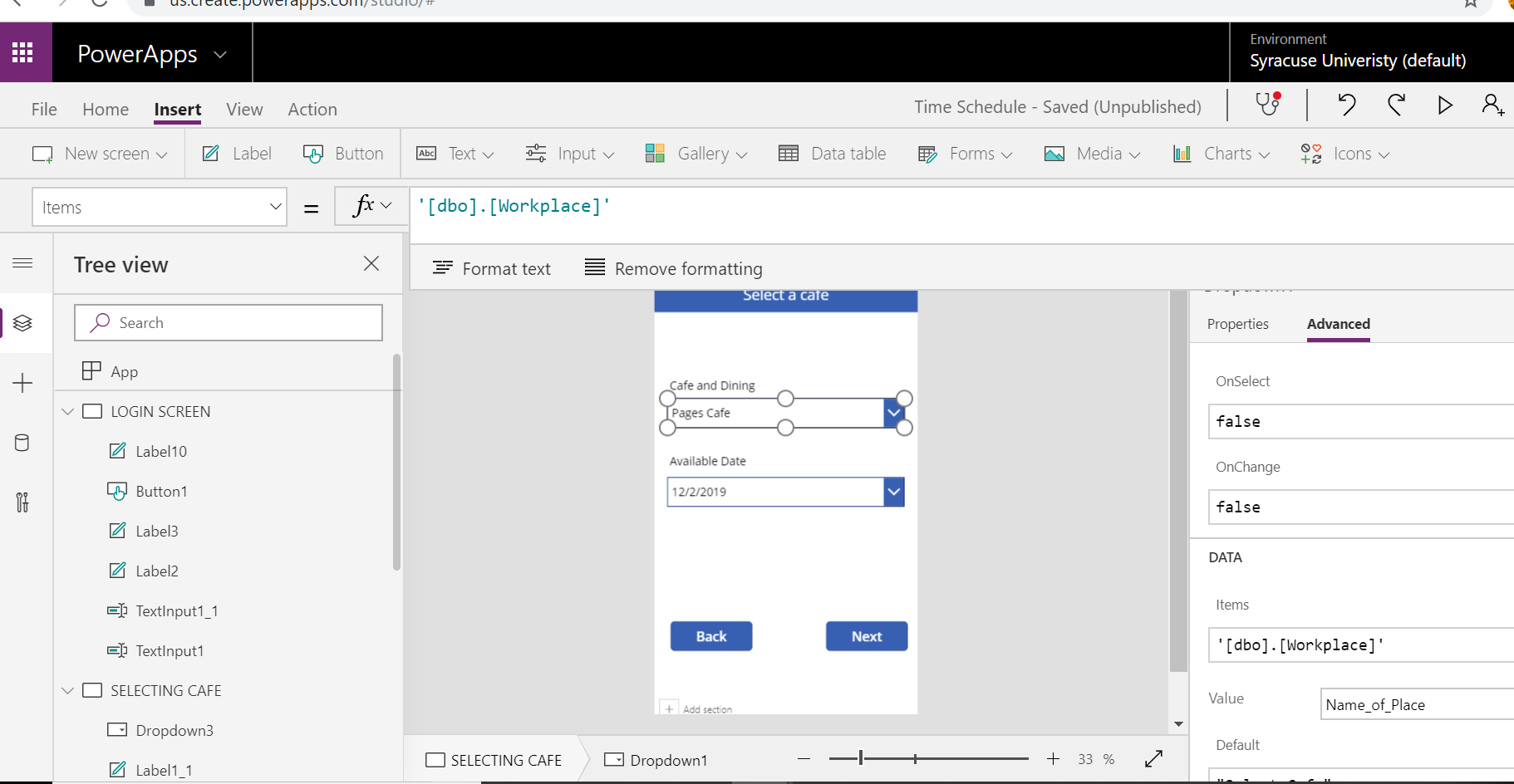


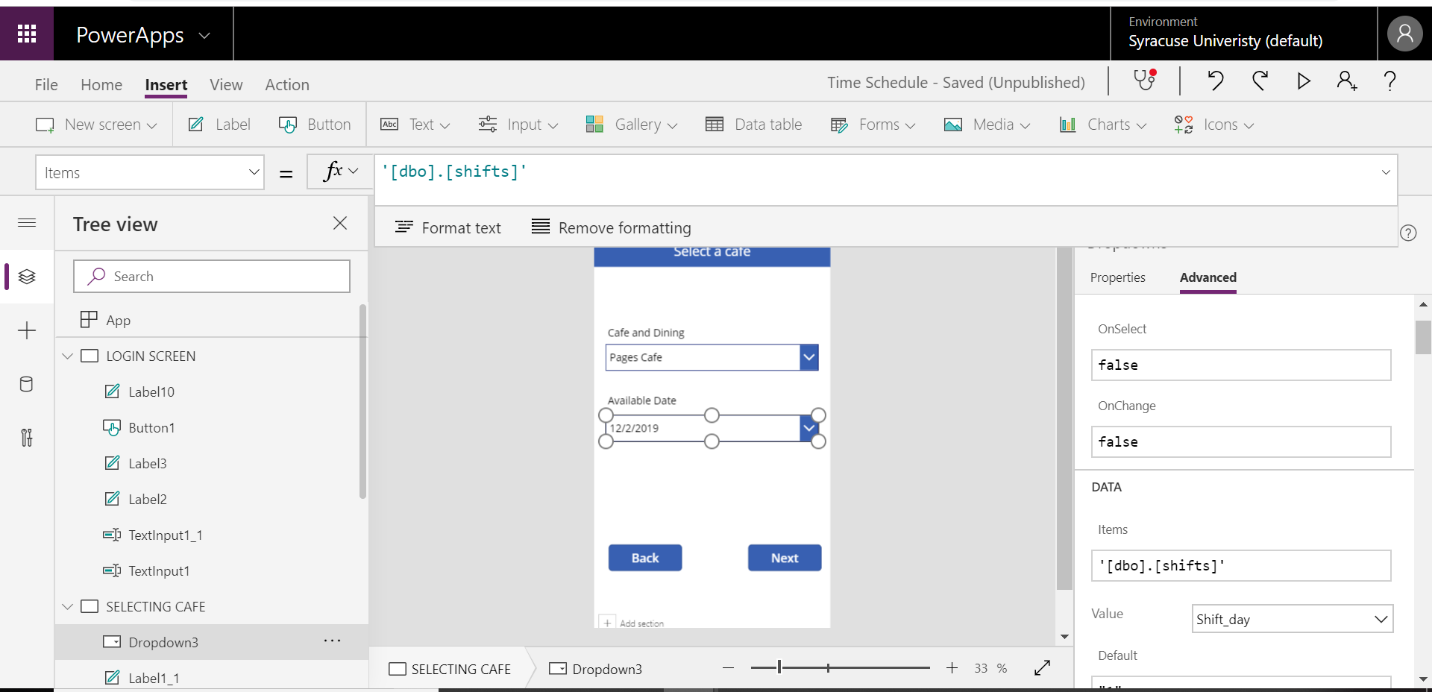


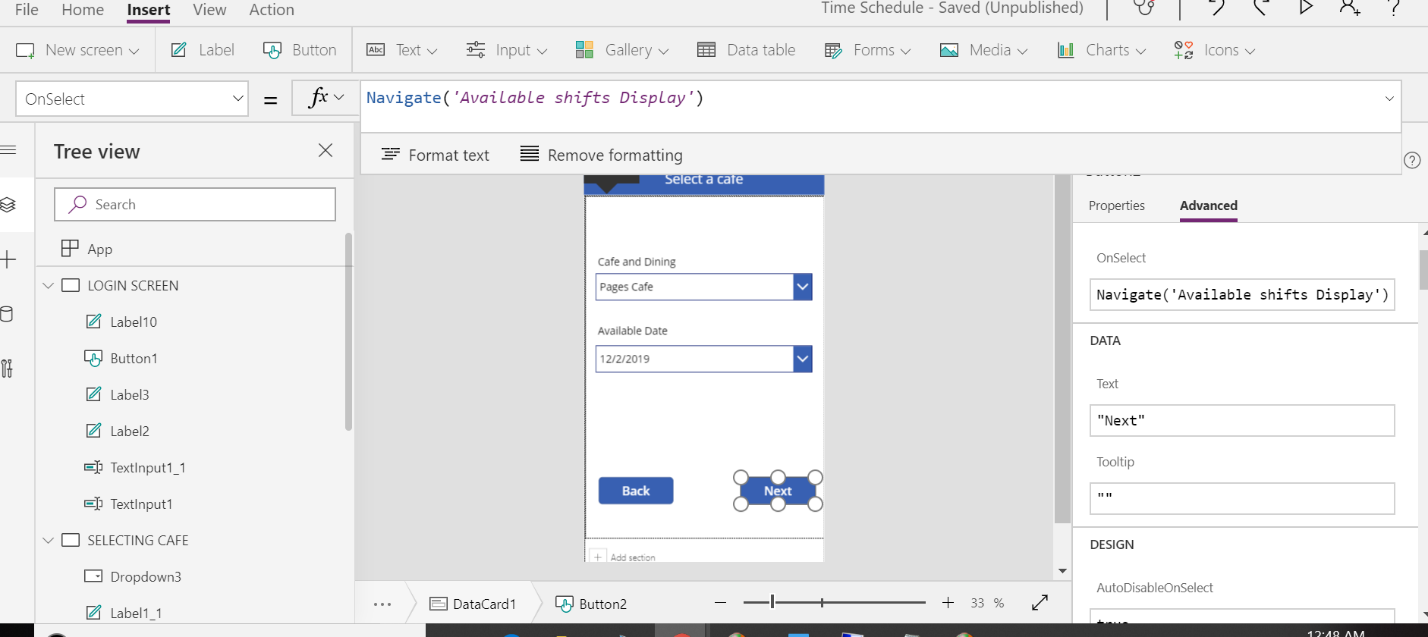


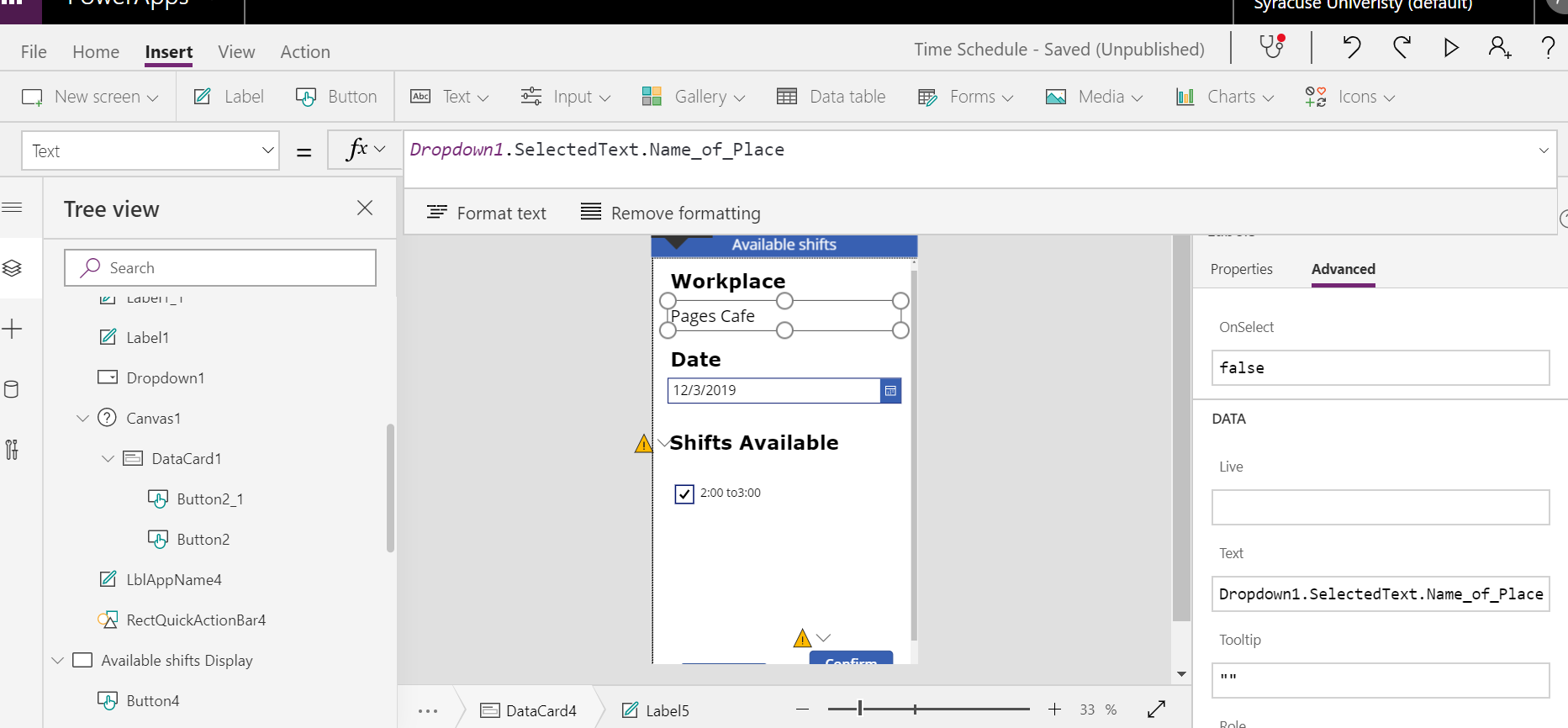
**Implementation in Power Apps:**

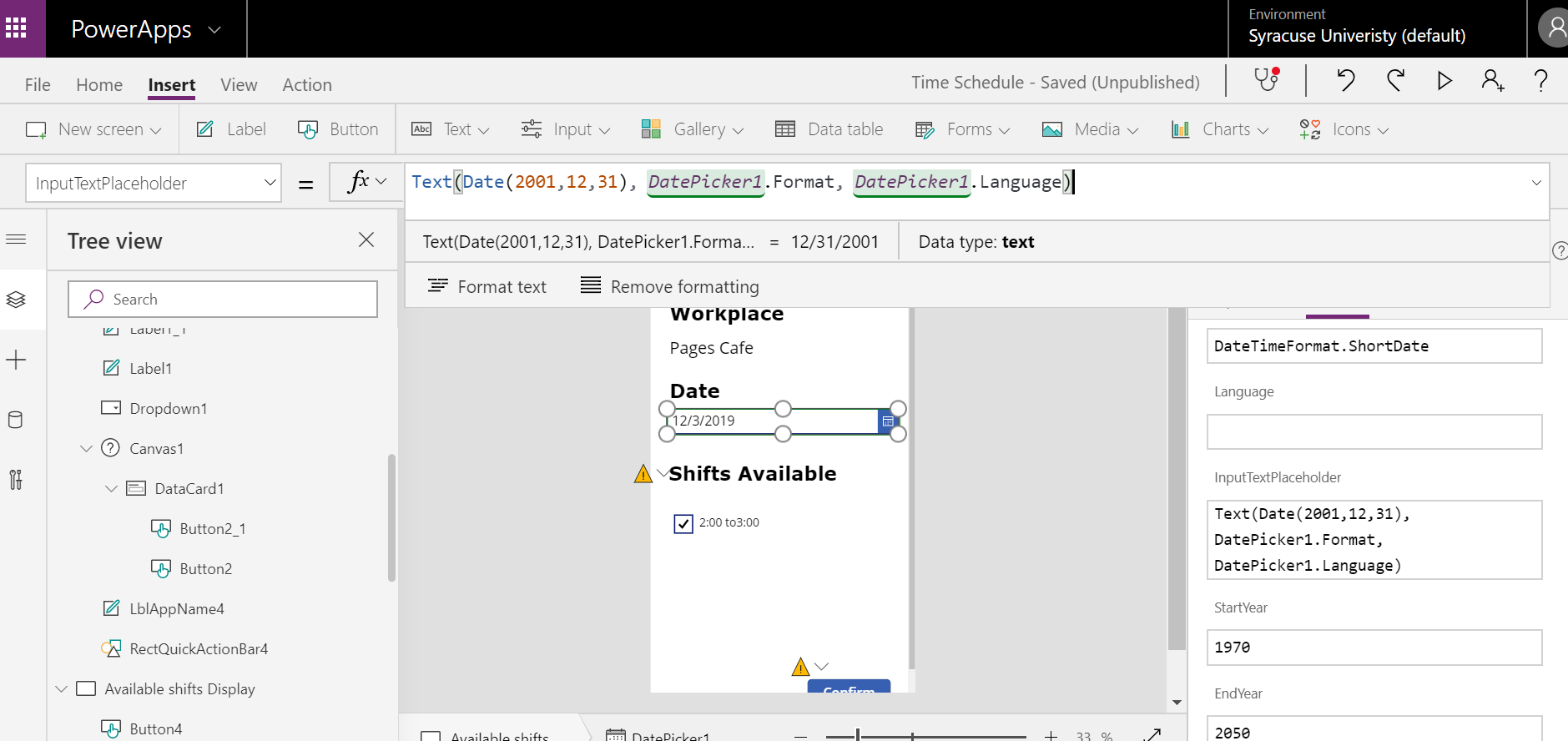


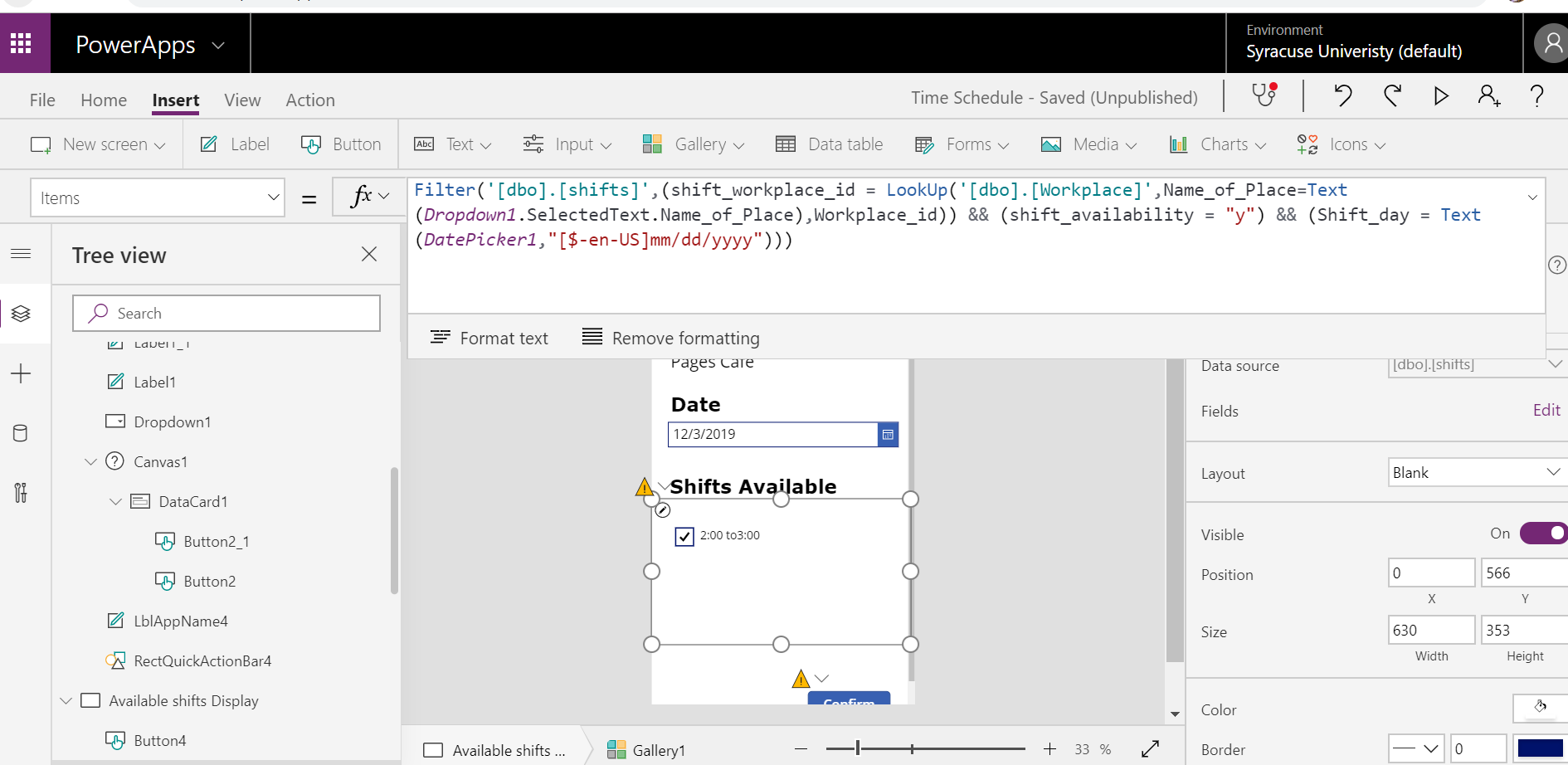


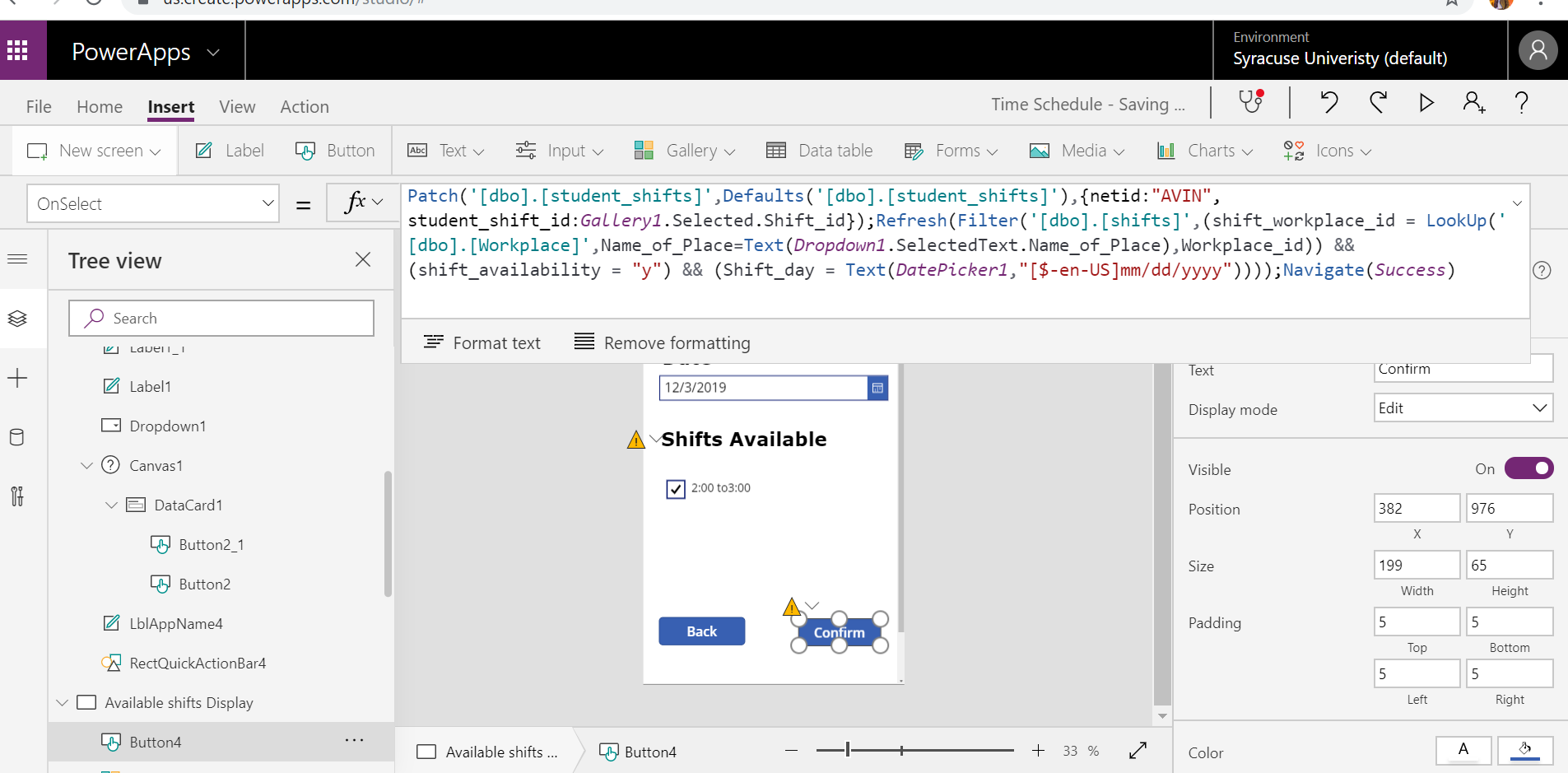












**LOG REPORT:**

**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

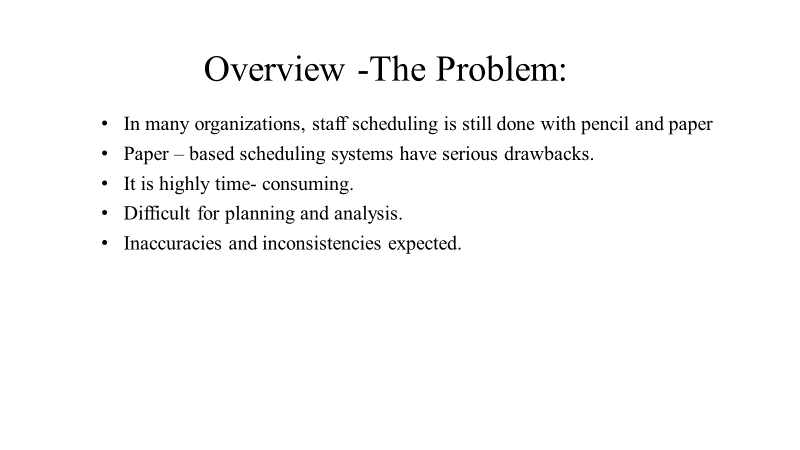
**Lavnish** - Made the presentation for the app

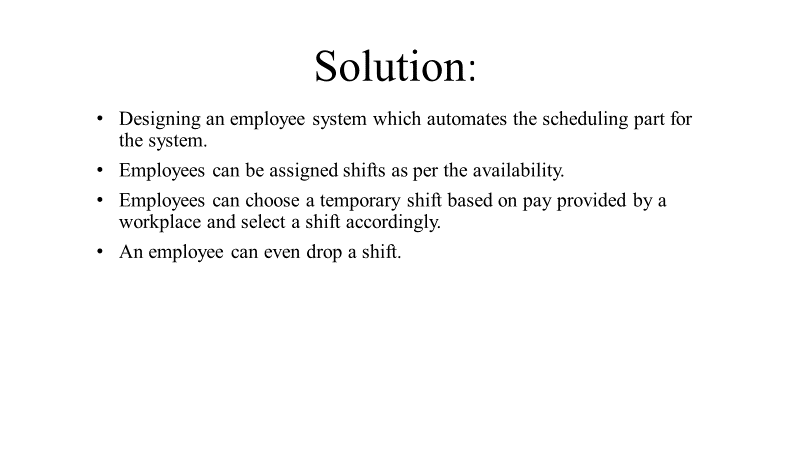
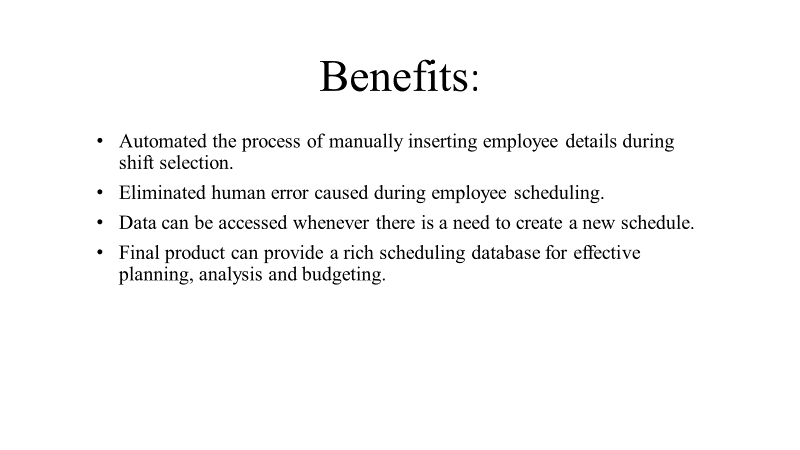
**1 December:**

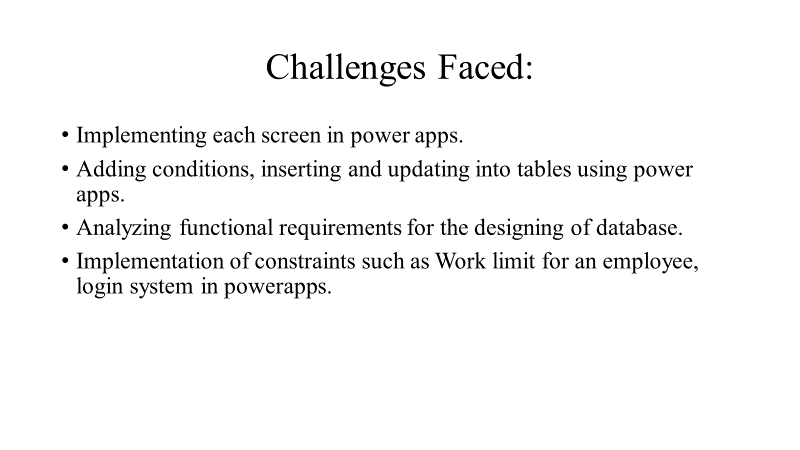
Team recorded a video where we explained the presentation and showed a demo of our app.

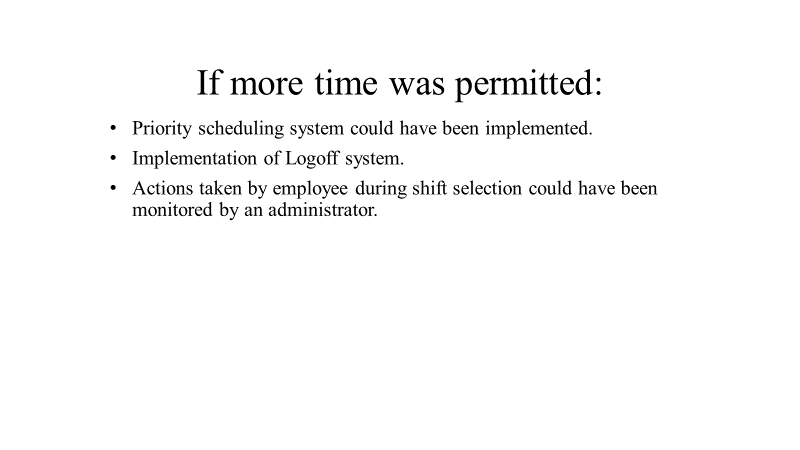
**Presentation:**









**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>

**PowerApps:**

<https://apps.powerapps.com/play/f75ee0f9-e479-4cce-9a25-d33f6655ee56?tenantId=4278a402-1a9e-4eb9-8414-ffb55a5fcf1e>