

Our project is a class management system for a school. The system does not require much; it uses Python Flask and HTML. The database software we used was supabase, it allows us to host the database on an online platform so we don't need to host it ourselves, it also gives a nice UI where we can interact with the database.

This is the SQL code:

```
-- WARNING: This schema is for context only and is not meant to be run.  
-- Table order and constraints may not be valid for execution.
```

```
CREATE TABLE public.Blackout Hours (  
    room_id bigint GENERATED ALWAYS AS IDENTITY NOT NULL,  
    start timestamp with time zone,  
    end timestamp with time zone,  
    reason text,  
    CONSTRAINT Blackout Hours_room_id_fkey FOREIGN KEY (room_id)  
    REFERENCES public.Room(room_id)  
);  
  
CREATE TABLE public.Building (  
    building_id text NOT NULL,  
    name text,  
    CONSTRAINT Building_pkey PRIMARY KEY (building_id)  
);  
  
CREATE TABLE public.Class Request (
```

```
request_id bigint GENERATED ALWAYS AS IDENTITY NOT NULL,  
section_id bigint NOT NULL,  
requester text,  
requested_start timestamp without time zone,  
requested_end timestamp without time zone,  
preferred_room text,  
status text NOT NULL DEFAULT 'unassigned'::text,  
CONSTRAINT Class Request_pkey PRIMARY KEY (request_id),  
CONSTRAINT Class Request_section_id_fkey FOREIGN KEY (section_id)  
REFERENCES public.Section(section_id)  
);
```

```
CREATE TABLE public.Course (  
course_id text NOT NULL,  
name text,  
department_id text,  
CONSTRAINT Course_pkey PRIMARY KEY (course_id),  
CONSTRAINT Course_department_id_fkey FOREIGN KEY (department_id)  
REFERENCES public.Department(department_id)  
);
```

```
CREATE TABLE public.Department (  
department_id text NOT NULL,  
name text,  
building_id text,
```

```
CONSTRAINT Department_pkey PRIMARY KEY (department_id),  
CONSTRAINT Department_building_id_fkey FOREIGN KEY (building_id)  
REFERENCES public.Building(building_id)  
);  
  
CREATE TABLE public.Equipment Type (  
    equip_id bigint GENERATED ALWAYS AS IDENTITY NOT NULL,  
    name text,  
    CONSTRAINT Equipment Type_pkey PRIMARY KEY (equip_id)  
);  
  
CREATE TABLE public.Request Equipment (  
    request_id bigint GENERATED ALWAYS AS IDENTITY NOT NULL,  
    room_id bigint,  
    equip_id bigint,  
    quantity bigint DEFAULT '1'::bigint,  
    CONSTRAINT Request Equipment_pkey PRIMARY KEY (request_id),  
    CONSTRAINT Request Equipment_room_id_fkey FOREIGN KEY (room_id)  
    REFERENCES public.Room(room_id),  
    CONSTRAINT Request Equipment_equip_id_fkey FOREIGN KEY (equip_id)  
    REFERENCES public.Equipment Type(equip_id),  
    CONSTRAINT Request Equipment_request_id_fkey FOREIGN KEY (request_id)  
    REFERENCES public.Class Request(request_id)  
);  
  
CREATE TABLE public.Room (
```

```
room_id bigint GENERATED ALWAYS AS IDENTITY NOT NULL,  
building_id text,  
room_num text,  
CONSTRAINT Room_pkey PRIMARY KEY (room_id),  
CONSTRAINT Room_building_id_fkey FOREIGN KEY (building_id) REFERENCES  
public.Building(building_id)  
);  
  
CREATE TABLE public.Room Assignment (  
assignment_id bigint GENERATED ALWAYS AS IDENTITY NOT NULL,  
room_id bigint,  
request_id bigint,  
start timestamp with time zone,  
end timestamp with time zone,  
status text,  
section_id bigint,  
CONSTRAINT Room Assignment_pkey PRIMARY KEY (assignment_id),  
CONSTRAINT Room Assignment_room_id_fkey FOREIGN KEY (room_id)  
REFERENCES public.Room(room_id),  
CONSTRAINT Room Assignment_section_id_fkey FOREIGN KEY (section_id)  
REFERENCES public.Section(section_id),  
CONSTRAINT Room Assignment_request_id_fkey FOREIGN KEY (request_id)  
REFERENCES public.Class Request(request_id)  
);
```

```
CREATE TABLE public.Room Equipment (
    room_id bigint GENERATED ALWAYS AS IDENTITY NOT NULL,
    equip_id bigint,
    quantity bigint DEFAULT '1'::bigint,
    CONSTRAINT Room Equipment_equip_id_fkey FOREIGN KEY (equip_id)
        REFERENCES public.Equipment Type(equip_id),
    CONSTRAINT Room Equipment_room_id_fkey FOREIGN KEY (room_id)
        REFERENCES public.Room(room_id)
);
```

```
CREATE TABLE public.Section (
    section_id bigint GENERATED ALWAYS AS IDENTITY NOT NULL,
    course_id text,
    instructor text,
    term text,
    CONSTRAINT Section_pkey PRIMARY KEY (section_id),
    CONSTRAINT Section_course_id_fkey FOREIGN KEY (course_id) REFERENCES
        public.Course(course_id)
);
```

Manuals:

Student: The student only needs to navigate the student page where there are drop down menus, class lets you pick the course, department, and time of day.

Secretary: This role only allows you to make requests for classes and edit said requests only. To make a request you navigate to the secretary page where you can fill in the boxes according to the requester's information, once you fill it out you submit it. If you need it to edit the request for any reason you can scroll down to access the existing request and change any information needed by clicking on the drop downs.

Admin: Admin can accept any incoming request, and manage existing classes. When you open the admin page you will see the requests. The admin can see the request and pick a room, if it's not available then there is an option for the class to be automatically assigned to the next available room. If you scroll down on the admin page you can see the existing room assignments and edit them. You can change the room, date and time.

Contribution:

Blake: Worked on UI and the Python backend as well as setting up supabase

Zedric: Worked on setting up the Supabase, Made slides, and filled out all info in the database