TABLE OF CONTENTS

1 UI Design Elements

1.1 Navigation

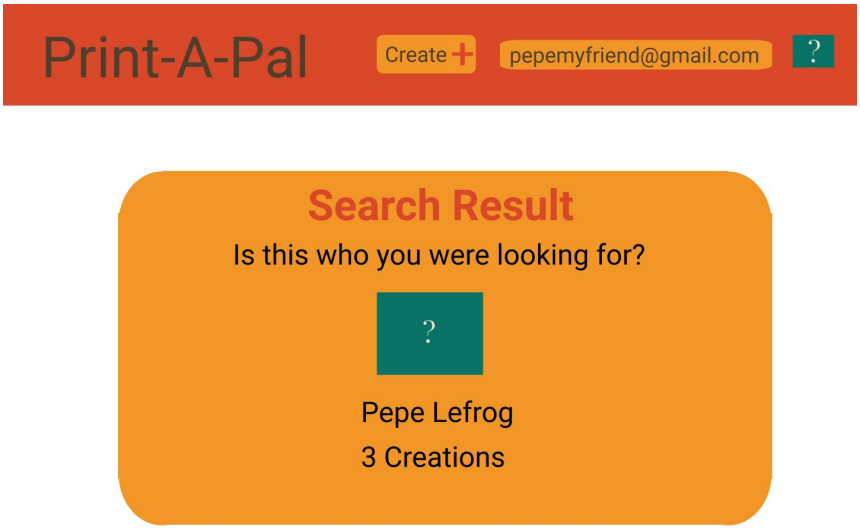

Figure 1: Image of the search result page

When first accessing the website, users are directed to the landing page. From here, they can choose to sign in or register their accounts. The blue avatar icon at the top right of the screen can also be clicked to bring you to the log in page, and the search bar can be used at any time to search for other accounts using the email address associated with their account.

When logged in, users are directed to their profile page, where they can edit their profile or change settings. Users can additionally browse their past creations, or click the Create + button to start a new creation residing at the top of the menu once logged in.

In the browsing page, you can view any of your past creations to start working on them again. Searching for users by email address brings you to the result page where a single user is displayed with their avatar, name and number of creations. You can then click their profile to go to their profile page and browse their creations.

In the creation menu, users can use the available tools to design their creation, and can save their progress and exit or submit the creation to be printed in the order page.

1.2 Landing and Registration Page

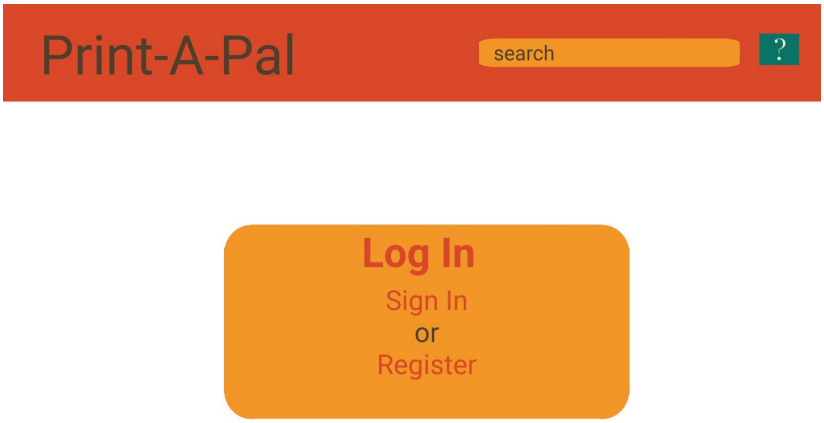

Figure 2: An image of the landing page/log in page

The landing page, if not signed in, gives you the option to log in, register for an account, or search for other users by entering their email in the search menu.

In the registration page, users can register for their account with Print-A-Pal, by entering their email, desired password, name, and age (optional).

1.3 Profile Page

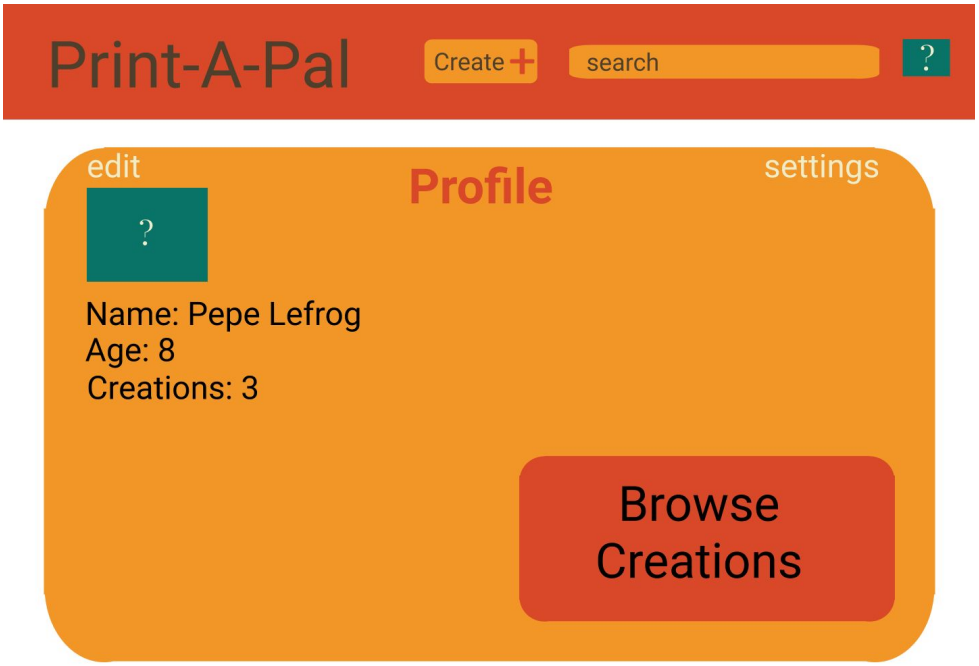

Figure 3: An image of the profile page

In the profile page, users are able to edit their profile, change their account settings, or click the Browse Creations button to browse the profile's creations. The name, age, and avatar are allowed to be changed in the edit profile section. The application's notifications, theme, and payment options are changeable from the account settings menu.

1.4 Browse Creations

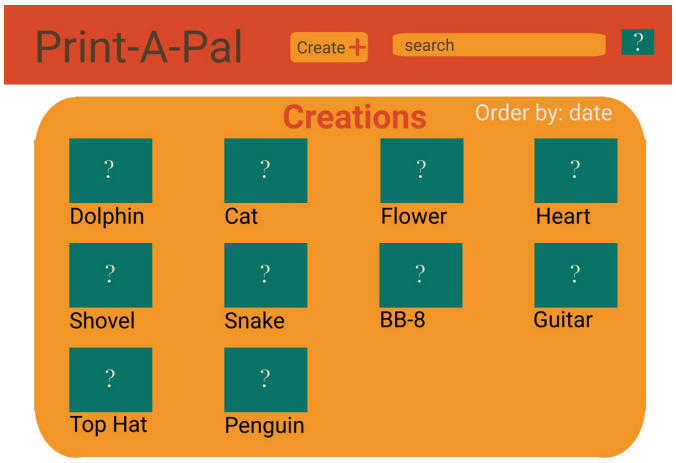

Figure 3: An image of the browse creations page

In the browse creations page, users can choose in which order to cycle through their past creations, giving them the option to view or edit any of their work. When browsing another user's creations, users are only allowed to view the rendering in different angles and not able to change anything.

1.5 Order Page

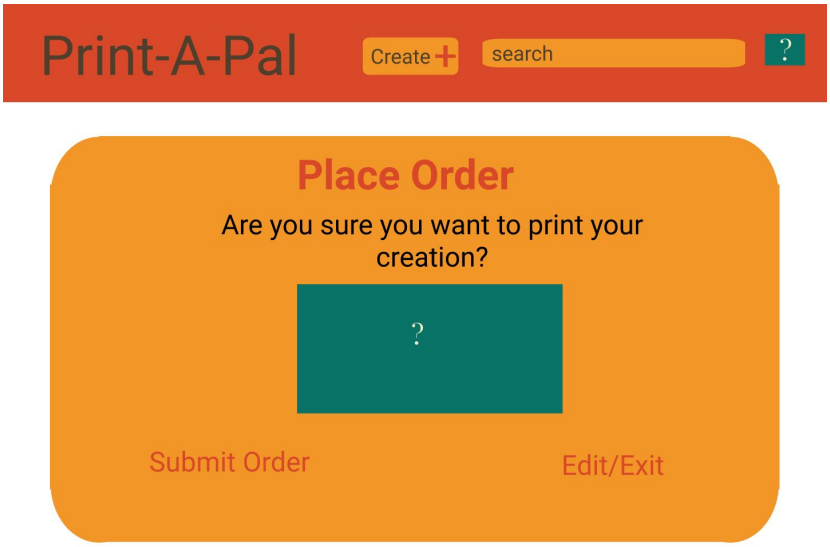

Figure 4: An image of the order page

In the order page, users are asked to confirm their choice of sending their work to be 3D printed, or go back to the creation page to edit their work further. Choosing to submit order takes users to the payment and delivery section which enables them to place the order for their creation to be 3D printed.

1.6 Creation page

In the creation page, users are given a variety of tools to work with to adjust and change the design of their creation. There are two main views within the creation page: the front view and side view. The front view is a 2D representation of the work space and where users do the majority of designing for their creation, drawing shapes onto the 2D plane. In the side view, users are given a 3D side angle of their drawing, where they can adjust the thickness and roundness of the edges of the shapes they have created in the 2D front view. The camera angle is adjustable to show different views of the image, unlike the 2D drawing view where users can only draw onto the 2D plane.

2 Module Decomposition

2.1 2D drawing module

Set of drawing operations for 2D shapes

- createCircle(radius)
    - Draws a circle of specified radius
- createRectangle(startX,startY,endX,endY)
    - Draws a rectangle with the specified vertices
- createLine(startX,startY,endX,endY)
    - Draws a line with specified vertices
- pencilDraw(startX,startY,enable_drag)
    - Draws on canvas like a pencil
- getcurrentMouseCoords(mousex,mousey)
    - Get current mouse position

2.2 Math Library for 3D space

Set of operations and objects for calculating 3D object primitives

- Objects
    - Vector(x,y,z)
        - Defines 3D vector object
    - Coord(x,y,z)
        - Defines 3D coordinate object
    - Plane(x,y,z,d)
        - Defines 3D plane object
- Operations
    - normalVector(vec1, vec2)
        - Calculates the normal vector given two vectors
    - vecMagnitude(vec)
        - Calculates the magnitude of a vector
    - normalizeVec(normalizeVector)
        - Returns normal form of vector
    - planeEq(coord1, coord2, coord3)
        - Given 3 coordinates return the equation of a plane

2.3 3D rendering module for planes and object

Set of operations and objects drawing 3D primitives, meshes and objects

- Objects
    - Mesh(length,drawfunc)
        - Creates a mesh by initalizing it’s array to the specified length and draws the mesh according to previously defined draw function
- Operations

- drawMesh(meshObject)
    - Iterates over the mesh object data structure and draws it
- createsMesh(typeOfMesh)
    - Creates and returns a mesh object that can be later used with drawMesh()
- drawTri(coord1, coord2, coord3)
    - Used to draw triangles for mesh objects

2.4 Custom 2D to 3D projection module

Set of operations and objects for calculating 3D world coordinates by taking the 2D image and "unprojecting" the values to return 3D world scene coordinates

- Objects
    - ViewMatrix and ProjectionMatrix
        - These objects are used to calculate the projection
- Operations
    - get2DPoint(coord1, ViewMatrix, ProjectionMatrix, length, width)
        - Returns the 2D screen point of 3D scene world
    - get3DPoint(coord1, ViewMatrix, ProjectionMatrix, length, width)
        - Returns the 3D world coordinate of a 2D projection point

2.5 Module for 3D mesh object to STL file

Set of operations and objects that are used to convert a mesh object to an STL file format for 3D printing purposes

- Operations
    - parseMesh(meshObject)
        - Goes through the mesh object list for the series of triangles and their respective face normal and returns STL file for the specified mesh object

2.6 User management module

Set of operations and objects that associates a user object with their creations. The objects in this sections are also representative of schema definition.

- Objects
    - User(id,name,username,password)
    - Object3D(id, object_type, project_id, file_id)
    - FIle(id, file_url)
    - Project(id, user_id)
- Operations
    - createNewUser(name,username,email,password,confirm_pass)

    - creates a new user object and saves it to the database
- deleteUser(id)
    - deletes the user from the database
- editUser(id, options = {})
    - grabs the user by the id and the options hash can be used to edit the Individual attributes the make up the User Object and save it to database
- returnUser(id)
    - grabs the user by the ID
- createNewFile()
    - creates a new File and saves the file_url to the database
- editFile(id, options = {})
    - grabs the file by the id and the options hash can be used to edit the Individual attributes the make up the User Object and save it to database
- deleteFile(id);
    - deletes the file from the database
- returnFile(id);
    - returns the file from the database according to its ID
- createNewProject(user_id);
    - creates a new Project specified by user_id and saves it to the database
- deleteProject(id)
    - deletes a project according to the ID
- returnProject(id)
    - returns a project according to the ID
- createNewObject3d(object_type, project_id, file_id)
    - creates a new object specfied by project_id, file_id and object_type and saves it to the database
- deleteObject(id)
    - delete the object from the database according to the ID
- returnObject(id)
    - returns the object from the database according to the ID
- editObject3D(id, options={})
    - grabs the file according to the ID and edits can specified the options where the key is the attribute and its associated value
- returnObject3d(id, options={})
    - returns object according to the ID
- readFile(file_url)
    - reads a file specified by the URL
- writeFile(file_url)
    - writes to a file specified by the URL

3 Technology Stack Breakdown

3.1 Frontend

We will be using the standard html/css/javascript stack with additional libraries to fully support the product's functionality. We will use jquery library for basic javascript. We will also use three.js which is a javascript library for WebGL. For styling purposes will use bootstrap for consistent styling throughout the web application

3.2 Backend

We will use ruby on rails framework since it has many operations within the framework already setup for starting up a web application. The framework will run off an nginx server. We will also use SQL database for information storage and retrieval and also C programming language for creating additional wrapper functions for the ruby on rails framework to support conversion from a 3D mesh object to a standard STL file.

3.3 Additional APIs

Wolfram Alpha library may be used for functions that are computationally intensive. Also we will use Amazon Web Services S3 module for storing files. The file URL will stored in the database. We will also use Heroku for hosting as it is already configured for Ruby on Rails Applications.

4 ER Diagram

The following diagram represents the database schema for the application

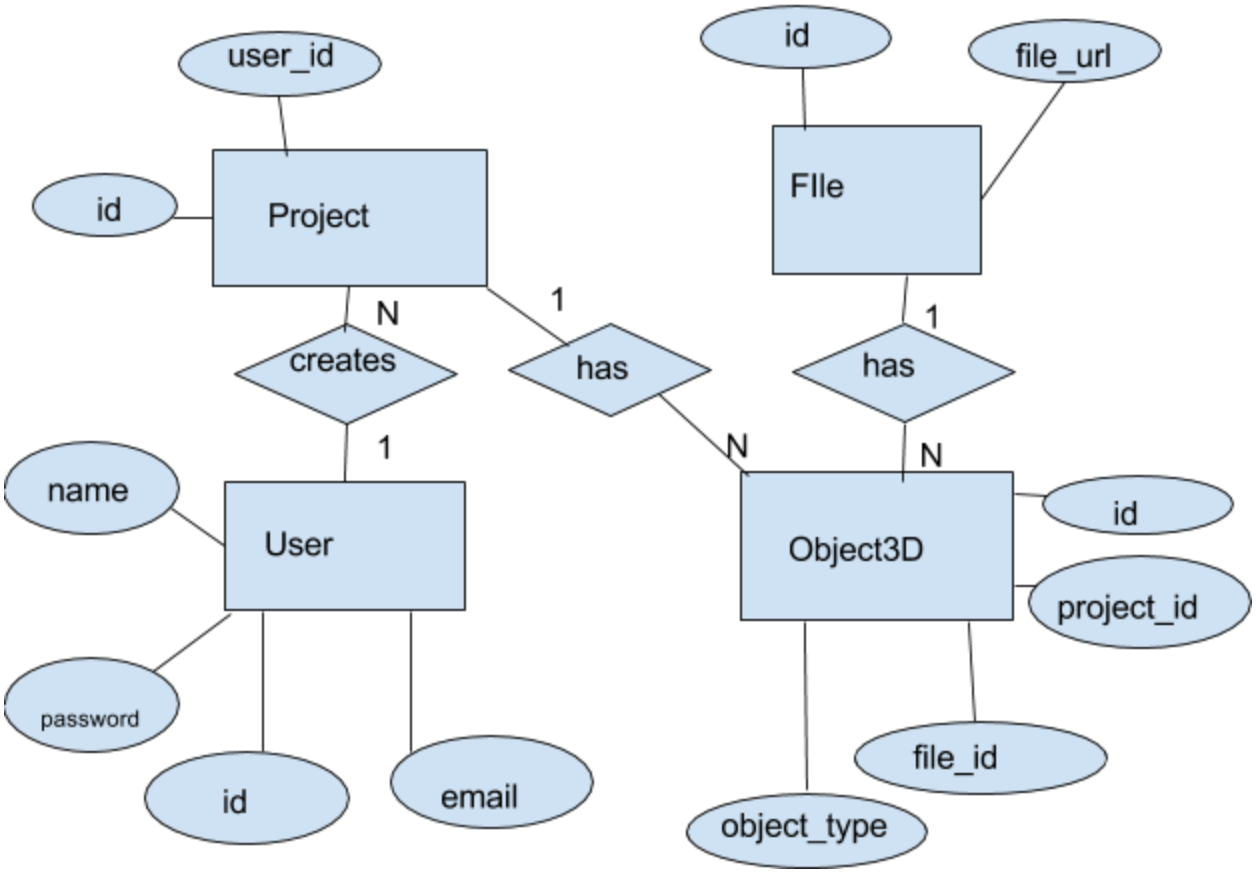

Figure 5: A Entity-Relationship diagram of Print-a-Pal