A

Project Report On

#### “STANDOUT 3D RESUME BUILDER”

SUBMITTED BY

### Mr. Arpit Katiyar

###### (Bachelor of Computer Application)

To,

University of Tilak Maharashtra Vidyapeeth (2023 - 2024)

**OM SAI INFOTECH**

## STANDOUT 3D RESUME BUILDER

OM SAI INFOTECH

101,SaiAnand Plaza, EktaSaphalya Society, Castlemill Circle, LBS Marg, Thane(w) T:022-25478222/ 8652552888 | E:

[castlemill.aptech@rediffmail.com](mailto:castlemill.aptech@rediffmail.com)

DATE :-17-05-2024 DIRECTOR RECOMMENDATION

To,

The Register,

Tilak Maharashtra University, Pune.

Subject: - Request approval of BCA III (SEM - VI) Project Report

We are recommending the report entitled - “ **STANDOUT 3D RESUME BUILDER** ” prepared **Mr. Arpit Katiyar.** As per partial fulfillment of university requirement for the award of **Bachelor of Computer Application (BCA III, SEM - VI)** degree of **TILAK MAHARASHTRA VIDYAPEETH, OM SAI INFOTECH.**

OM SAI INFOTECH

101, SaiAnand Plaza, EktaSaphalya Society, Castlemill Circle, LBS Marg, Thane(w) T:022-25478222/ 8652552888 | E:

[castlemill.aptech@rediffmail.com](mailto:castlemill.aptech@rediffmail.com)

DATE :-17-05-2024

CERTIFICATE

This to certify that **Mr. Arpit Katiyar** of **T.Y.BCA** have satisfactorily completed the project entitled, **“STANDOUT 3D RESUME BUILDER”** and submitted in partial fulfillment of requirement for the award of BCA III year (Semester – VI) during the academic year 2023 – 2024 is record of students own work carried out under my supervision and Guidance.

Mrs. Anjali Phatak

(Director) (Project guide)

External Sign:

## ACKNOWLEDGEMENT

#### ACKNOWLEDGEMENT

###### We are very glad that we have successfully completed our assignment on time. We appreciate all the helpers for helping out along the way of this development. We thank all of them for their helpful attitude.

We would like to **express our sincere gratitude** to our lecturer, **Mrs. Anjali Phatak**, for her guidance throughout this project. Her instruction on various **software development methodologies** and their **benefits**, as well as the fundamentals of **UML Design**, provided a strong foundation for our work.

###### We are also **thankful** to everyone who supported us in completing this project. This includes the faculty of our department who assisted with our research needs.

We are **proud** to have successfully finished this project **on schedule**. We appreciate the assistance we received from **all those who helped us** along the way, particularly their **positive and supportive attitude**.

INDEX

**INDEX**

|  |  |  |
| --- | --- | --- |
| Sr.no | TITLE | Page No. |
| 1 | Introduction | [1](#_bookmark0) |
| 2 | Project Analysis | [3](#_bookmark1) |
| 3 | System Analysis | [8](#_bookmark2) |
| 4 | System design   * 1. Form Design   2. Flow chart   3. UML Diagram   4. E-R Diagram   5. Use case Diagram   6. Sequence Diagram   7. Class Diagram   8. Activity Diagram | [11](#_bookmark3) |
| 5 | Data Dictionary | [39](#_bookmark4) |
| 6 | Implementation of Project 6.1)Code  6.2) Snapshots/ Input & Output Design | [42](#_bookmark5) |
| 7 | Project Testing | [105](#_bookmark6) |
| 8 | Conclusion | [116](#_bookmark7) |
| 9 | Future Enhancement | [120](#_bookmark8) |
| 10 | Bibliography | [125](#_bookmark9) |

# INTRODUCTION

#### INTRODUCTION

In the rapidly evolving landscape of job hunting, StandOut emerges as a innovative solution, promising to reshape the traditional approach to resume building. By harnessing the power of cutting-edge 3D visualization, StandOut offers job seekers a dynamic platform to create immersive resumes. StandOut enables individuals to showcase their skills, achievements, and personal brand in a visually captivating manner.

Because of lack of individuality and inability to effectively convey one's full potential, even the most qualified candidates struggle to make an impact. Recognizing this challenge, StandOut steps in to provide a solution. Through its interactive 3D templates, users can tailor their resumes to reflect their personality and career aspirations.

In conclusion, StandOut marks a paradigm shift in self- presentation for job seekers. With StandOut, countless others have the opportunity to elevate their resumes, making them truly unforgettable in the eyes of potential employers.

- Arpit Katiyar

**PROJECT ANALYSIS**

1. **PROJECT ANALYSIS**
   1. **Project-Profile**

|  |  |
| --- | --- |
| **Project title** | **StandOut 3D Resume Builder** |
| **Objectives** | To provides a distinctive and engaging platform for building interactive 3D resumes. |
| **Front End Tool** | HTML, CSS, JavaScript, THREE.js |
| **Back End Tool** | Firebase Authentication, Firestore Database, Node.js, Vite.js |
| **Type Of Application** | Web Application |
| **Internal Guide** | **Mrs. Anjali Phatak** |

* 1. **Existing System**

Existing System (Traditional Resumes)

In the existing system of job applications, traditional resumes are the norm. These resumes are typically two- dimensional documents in formats like DOC or PDF. However, they come with several limitations:

* **Lack of Individuality:** Traditional resumes tend to lack personality and fail to represent the unique qualities of a candidate.
* **Limited Information Depth:** Due to space restrictions, traditional resumes often cannot adequately showcase a candidate's full range of skills and accomplishments.
* **Ineffective Communication:** Static text makes it challenging for candidates to capture the attention of recruiters and make a lasting impression.

To address these limitations and enhance the effectiveness of job applications, there is a need for innovative approaches that go beyond traditional resumes

* 1. **Need For A New System**

##### Need for a New System (StandOut)

* **Competitive Job Market**: The current job market demands candidates to stand out amidst fierce competition
* **Ineffective Traditional Resumes**: Traditional resumes often fail to effectively communicate a candidate's unique value proposition, resulting in them getting lost among numerous applications.
* **Introduction of StandOut**: StandOut addresses these limitations by offering a revolutionary 3D resume builder.
* **Visual and Interactive Experience**: StandOut empowers job seekers to create visually captivating and interactive experiences that effectively showcase their skills, achievements, and personal brand.
* **Breaking Through the Noise**: By leveraging StandOut, candidates can break through the noise of traditional applications and leave a memorable impression on recruiters, increasing their chances of securing coveted opportunities.

#### FUNCTIONALITIES

Seamless User Experience:

* **Simple Sign-up**: Register effortlessly with just your email and password, prioritizing ease over complex processes.
* **Intuitive 3D Builder**: Craft your 3D resume effortlessly using our user-friendly interface, guiding you through creating a stunning presentation of your skills and achievements.
* **Interactive Detail Input**: Input resume details directly into the immersive 3D environment, streamlining the process with engaging and straightforward elements.

Effortless Sharing:

* **Link-Based Sharing**: Generate a unique shareable link for your dynamic 3D resume, eliminating the need for emailing bulky documents.
* **Frictionless Access**: Recipients can access your resume without downloads or installations, viewing it directly from their browser for enhanced convenience.
* **Real-Time Updates**: Keep your resume current with instant updates reflected in the shared link, enabling you to adapt to new opportunities effectively.

# SYSTEM ANALYSIS

#### SYSTEM ANALYSIS

* 1. **Hardware Requirement(Minimum)**

|  |  |
| --- | --- |
| **Component** | **Specification** |
| Processor | AMD or higher |
| RAM | 512MB or more |
| Hard Disk | 10GB or more |
| Devices | Laptop, PC, Smartphones, Tablets, or any device capable of running a web browser |
| Keyboard | Normal or Multimedia |
| Mouse | Compatible Mouse |

* 1. **Software Requirement(Minimum)**

|  |  |
| --- | --- |
| **Component** | **Specification** |
| Operating System | Windows 10, Windows 8, Android, MacOS, or any operating system compatible with modern  web browsers |
| Browser | Brave, Chrome, Edge, Firefox, Safari, or any modern browser with HTML5 and CSS3 support |
| Internet Connection | Broadband connection recommended for optimal performance |
| Screen Resolution | 400x700 or higher |
| Plugins | No additional plugins required |

# SYSTEM DESIGN

#### 1 FORM DESIGN

In the 3D resume builder project, various forms play pivotal roles in facilitating user interactions and managing profile information effectively.

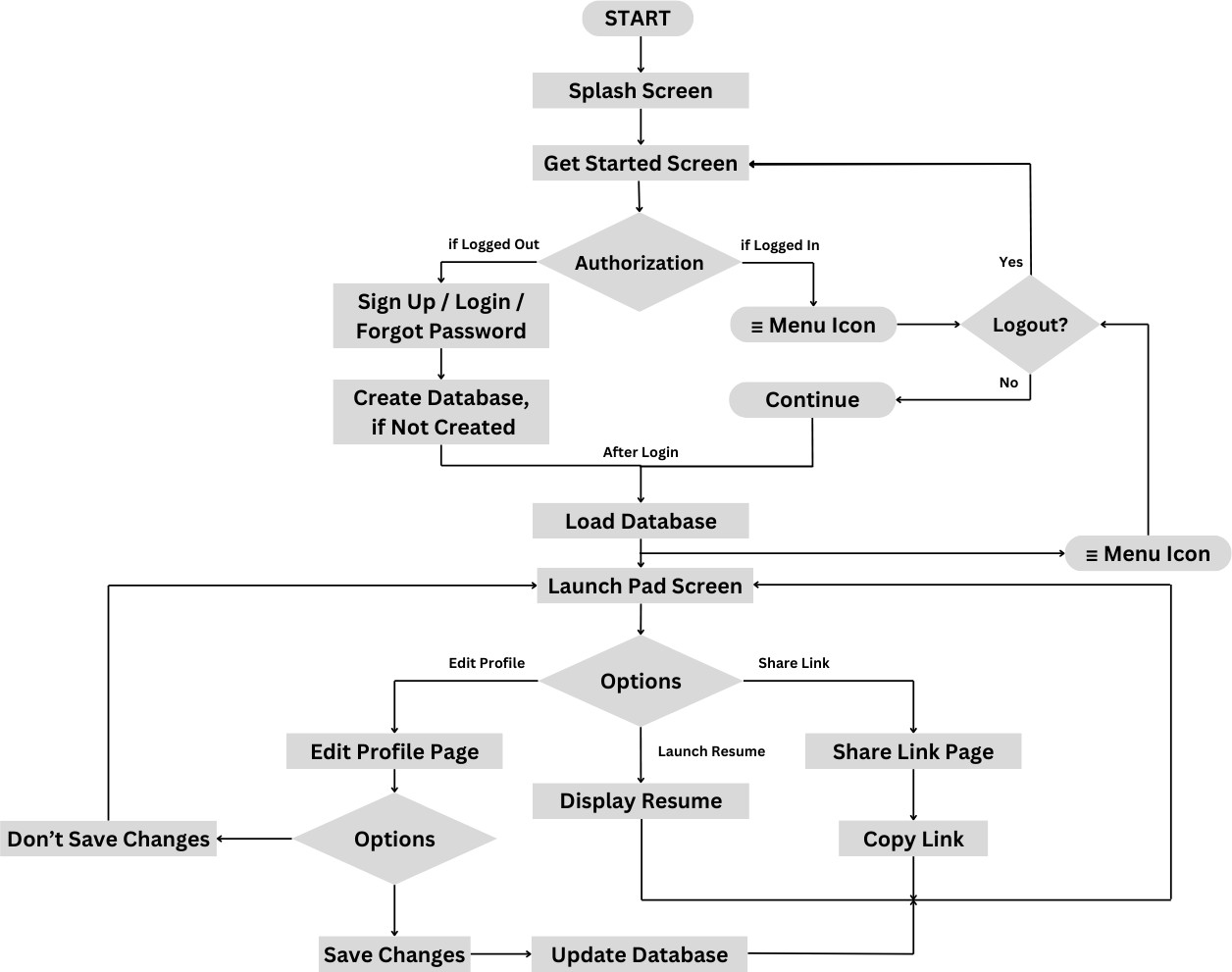
1. **Sign-Up Form**: This initial form prompts users to create an account by providing their email address and setting up a password. Upon submission, a verification link is sent to the provided email for authentication. Once verified, users gain access to the platform.
2. **Login Form**: Subsequently, users can log in to their accounts using the same credentials provided during sign- up - their email and password. This ensures secure access to their profile and resume-building functionalities.
3. **Forgot Password Form**: In the event of a forgotten password, users can utilize the "Forgot Password" form. Here, they input their registered email address, triggering a verification email containing instructions to reset their password. This process ensures account security and user convenience.
4. **Resume Profile Forms**: These forms cater to the detailed construction of a user's resume profile. They encompass various sections such as introduction, work experience,

skills, qualifications, personal information, and more. Each section typically consists of fields for title, subtitle, and detailed information, totalling to approximately 8 sections per form. Users can comprehensively tailor their resume profiles to showcase their professional attributes effectively.

1. **Shareable Link**: Upon completion, users can generate a shareable link for their 3D resume profile. This feature facilitates easy sharing with potential employers, recruiters, or other interested parties, enhancing the reach and accessibility of the user's professional portfolio.

Through these meticulously designed forms, the 3D resume builder project ensures seamless user experiences and robust management of profile information, empowering individuals to present their professional identities with confidence and clarity.

#### FLOW CHART:

****

###### The flowchart provided outlines the operational process of StandOut 3D Resume Builder.

Here's a detailed breakdown of each step, presented in a coherent and organized manner:

* + - **Start Point**: The process initiates from the designated START point.
    - **Splash Screen**: Users are greeted with a splash screen upon launching the application, setting the initial tone.
    - **Get Started Screen**: Users progress to the "Get Started" screen, initiating the journey.
    - **Authorization Check**: The application verifies user authorization status.
      * **If Logged Out**: Users are presented with options to either Sign Up, Login, or Reset Password, catering to various user scenarios.
      * **If Logged In**: Authenticated users seamlessly transition to the main menu, streamlining access to key features.
    - **Database Management**: The application dynamically manages the database based on user actions.
      * **Create/Load Database**: If no database exists, the system initializes one. Otherwise, it loads the existing database, ensuring data integrity.
    - **Launch Pad Screen**: Users arrive at the central interface, offering essential functionalities.
    - **Profile Editing**: Users can opt to edit their profiles, facilitating customization and updates.
      * **Save Changes**: Upon editing, users have the option to save changes, ensuring data accuracy and persistence.
      * **Discard Changes**: Alternatively, users can opt not to save changes, reverting to the previous state.
    - **Share Link**: Users can navigate to the share link feature, enabling easy dissemination of their resume.
      * **Display Resume**: Users can choose to display their resume directly from the application.
      * **Copy Link**: Alternatively, users can copy the shareable link, facilitating effortless sharing across platforms.

This meticulously designed flowchart delineates a logical progression of user interactions within the StandOut 3D Resume Builder application. It incorporates decision points and essential functionalities, offering users a seamless experience from initiation to resume sharing.

* 1. **UML Diagrams:**

Unified Modeling Language (UML) Section Overview:

In this section of the black book, various UML diagrams have been meticulously crafted to elucidate the structural and behavioral aspects of the StandOut 3D Resume Builder application.

UML serves as a standardized modeling language for visualizing, specifying, constructing, and documenting the system's architecture and functionalities.

The diagrams provided offer comprehensive insights into the application's design, facilitating better understanding and communication among stakeholders.

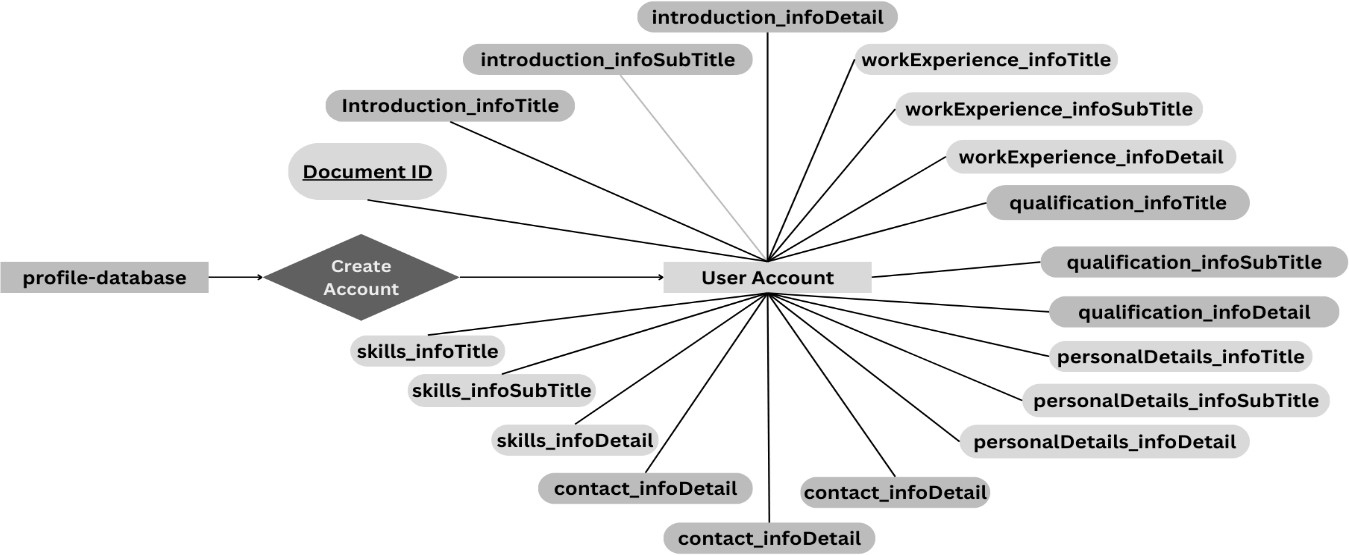
Below are the key UML diagrams featured in this section:

* + - **Use Case Diagram**: Illustrates the interactions between users and the system, identifying various use cases and their relationships.
    - **Sequence Diagram**: Presents the chronological sequence of interactions between system components or actors, depicting the flow of messages.
    - **Class Diagram**: Represents the static structure of the system, depicting classes, their attributes, methods, and relationships.
    - **Activity Diagram**: Depicts the flow of activities within the system, illustrating the sequence of actions and decision points.

These UML diagrams collectively provide a comprehensive overview of the StandOut 3D Resume Builder application, aiding in its analysis, design, and development phases.

#### E-R DIAGRAM:

###### The ER diagram provided offers insight into the database structure of the StandOut 3D Resume Builder application.



Below is a comprehensive breakdown of its components and their significance:

* + - **User Account Entity**: At the core of the diagram lies the User Account entity, serving as the focal point for user profiles within the application.
    - **Attributes**: Associated with the User Account entity are various attributes, each representing distinct sections of a resume:
      * **Introduction Info**: Comprising Title, Subtitle, and Detail, this section provides a succinct overview of the user.
      * **Work Experience**: Structured similarly to Introduction Info, this attribute delineates the user's professional background.

|  |  |  |  |
| --- | --- | --- | --- |
| * **Qualification**: Analogous to | | Work | Experience, |
| Qualification outlines  achievements. | the | user's | educational |

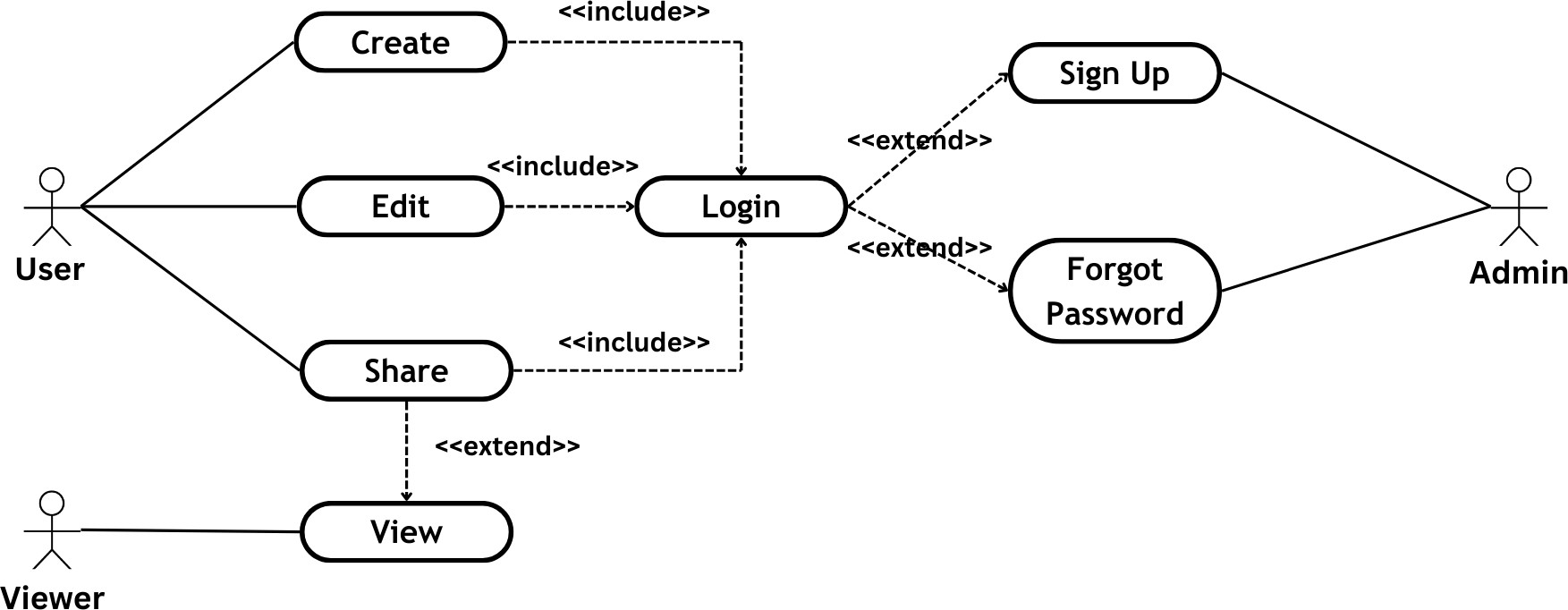
* + - * **Personal Details**: Contains personal information, including name and address, segmented into Title, Subtitle, and Detail.
      * **Skills**: Lists the user's proficiencies, following the same subdivided format.
      * **Contact Info**: Holds the user's contact details without further subdivision.
    - **Create Account Process**: Indicated by the "Create Account" process, this signifies the initiation of a User Account creation within the profile database.
    - **Document ID**: Each user entry may be associated with a unique Document ID, facilitating identification and retrieval within the database.
    - **Relationships**: The diagram illustrates one-to-many relationships between the User Account entity and its attributes, indicating that a user can have multiple entries for work experience, skills, etc.

This ER diagram serves as a visual representation of the database architecture underlying the StandOut 3D Resume Builder application. It elucidates the interconnections between different data components and elucidates how user information is organized around the central user profile, enhancing comprehension of the system's data structure.

#### USE CASE DIAGRAM:

**Use Case Diagram Overview:**

The Use Case Diagram presented in this section illustrates the interactions between various actors - User, Admin, and Viewer - and the corresponding functionalities within the StandOut 3D Resume Builder application.



This Use Case Diagram serves as a visual representation of the system's high-level behaviour, offering valuable insights into user interactions and system functionalities within the StandOut 3D Resume Builder application.

Below is a breakdown of the key components and relationships depicted in the diagram:

Actors:

* + - **User**: Represents individuals actively engaged in interacting with the system, including creating, editing, and sharing content.
    - **Admin**: Denotes system administrators or privileged users responsible for managing user accounts and assisting with account-related issues.
    - **Viewer**: Represents individuals who passively view content within the system without actively participating in its creation or management.

Use Cases:

* + - **Create**: Enables Users to generate new content within the system, such as crafting a 3D resume or profile.
    - **Edit**: Allows Users to modify existing content, ensuring flexibility in updating their profiles or resumes.
    - **Login**: Facilitates User authentication, granting access to restricted features and personalized content.
    - **Share**: Empowers Users to distribute their content to others, extending their professional reach and networking capabilities.
    - **View**: Permits Users to access and peruse shared content from other Users, fostering collaboration and inspiration.
    - **Sign Up**: Provides Admins with the capability to register new Users, expanding the user base and fostering growth.
    - **Forgot Password**: Equips Admins with tools to assist Users who have forgotten their login credentials, ensuring seamless access to the platform.

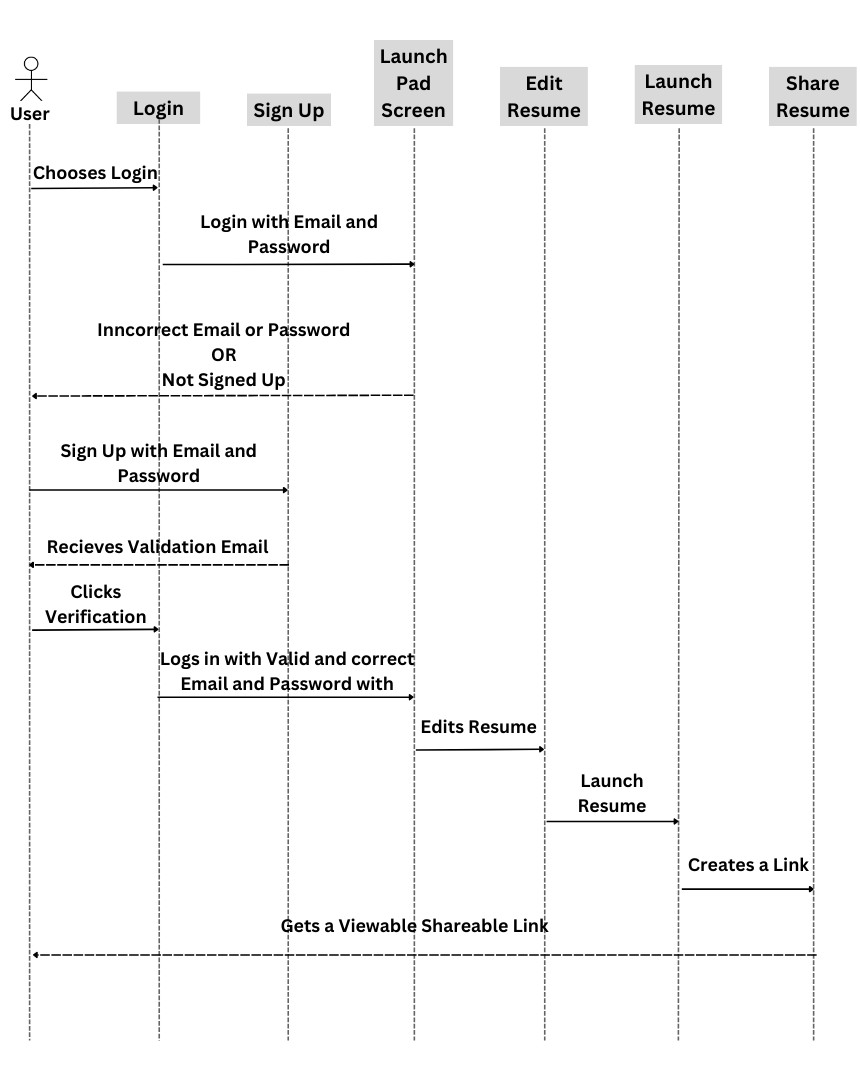
Relationships:

* + - **<includes>**: Signifies that certain use cases, such as Create, Edit, and Share, encompass the Login process, necessitating user authentication for access.
    - **<extends>**: Indicates optional or conditional functionality, as exemplified by View extending from Share, offering Users the opportunity to explore shared content at their discretion.

Detailed Explanation:

* + - **User**: Empowered to create, edit, share, and view content, ensuring a dynamic and engaging user experience tailored to individual preferences.
    - **Admin**: Equipped with administrative privileges, enabling them to oversee user registration and provide assistance to Users encountering login difficulties.

#### SEQUENCE DIAGRAM:



Sequence diagram and collaboration diagram are called Interaction Diagrams. An interaction diagram shows an interaction, consisting of set of objects and their relationship including the messages that may be dispatched among them.

A sequence diagram is an introduction that empathizes the time ordering of messages. Graphically a sequence diagram is a table that shows objects arranged along the X-axis and messages ordered in increasing time along the Y-axis

##### Sequence Diagram Overview:

The Sequence Diagram presented here illustrates the sequential interactions between users and system components within the StandOut 3D Resume Builder application. This diagram delineates the process flow for users as they engage with various resume-related functionalities, including login, sign up, editing, launching, and sharing. Below is a detailed explanation of the depicted sequence:

Detailed Explanation:

**User Interaction:**

1. **Login or Sign Up Decision:** Users initiate the interaction by choosing to either log in with existing credentials or sign up for a new account.
   * **If Login:** Users input their Email and Password for authentication.
     + If the credentials are incorrect or the user is not registered, they are prompted to sign up.
   * **If Sign Up:** Users provide their Email and Password for registration.
     + Upon submission, users receive a Validation Email for account verification.
     + Users click the validation link within the email to verify their account.

**Post-Authentication Actions:** 2. **Edit Resume:** Following successful authentication, users access the Launch Pad Screen to edit their resume.

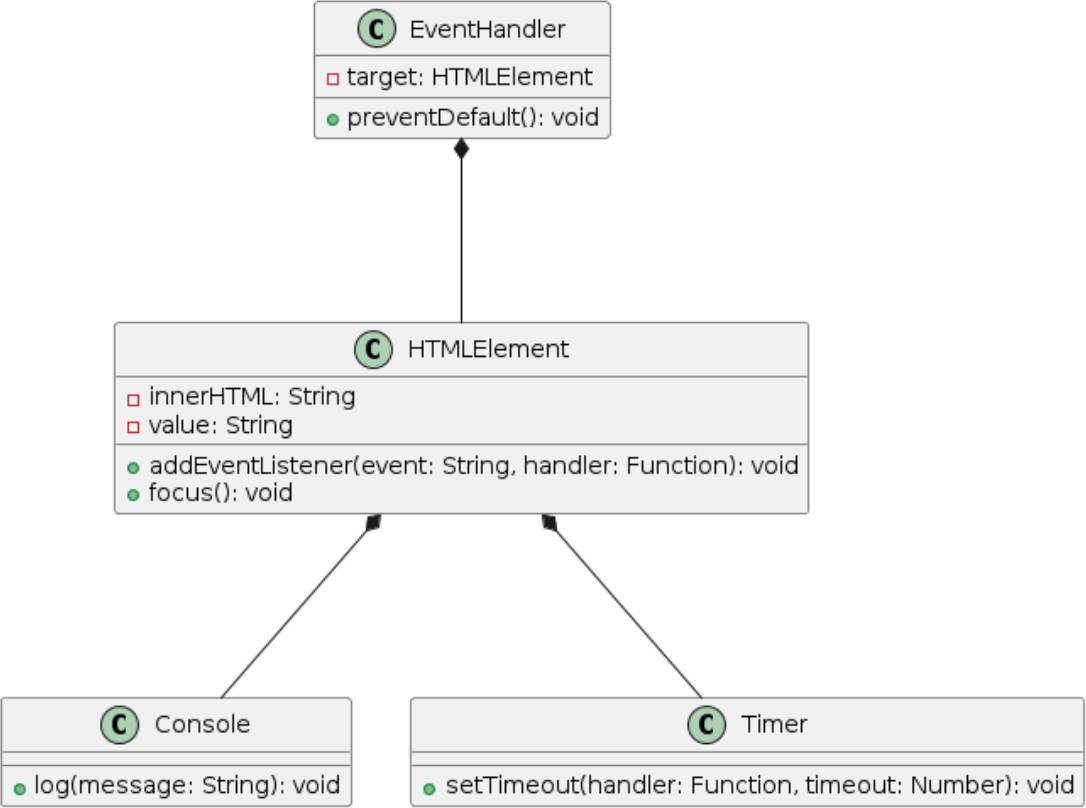
1. **Launch Resume:** Upon completion of editing, users initiate the process to launch their resume.
   * The system generates a Viewable Shareable Link for the resume, enabling users to distribute it seamlessly.

Conclusion:

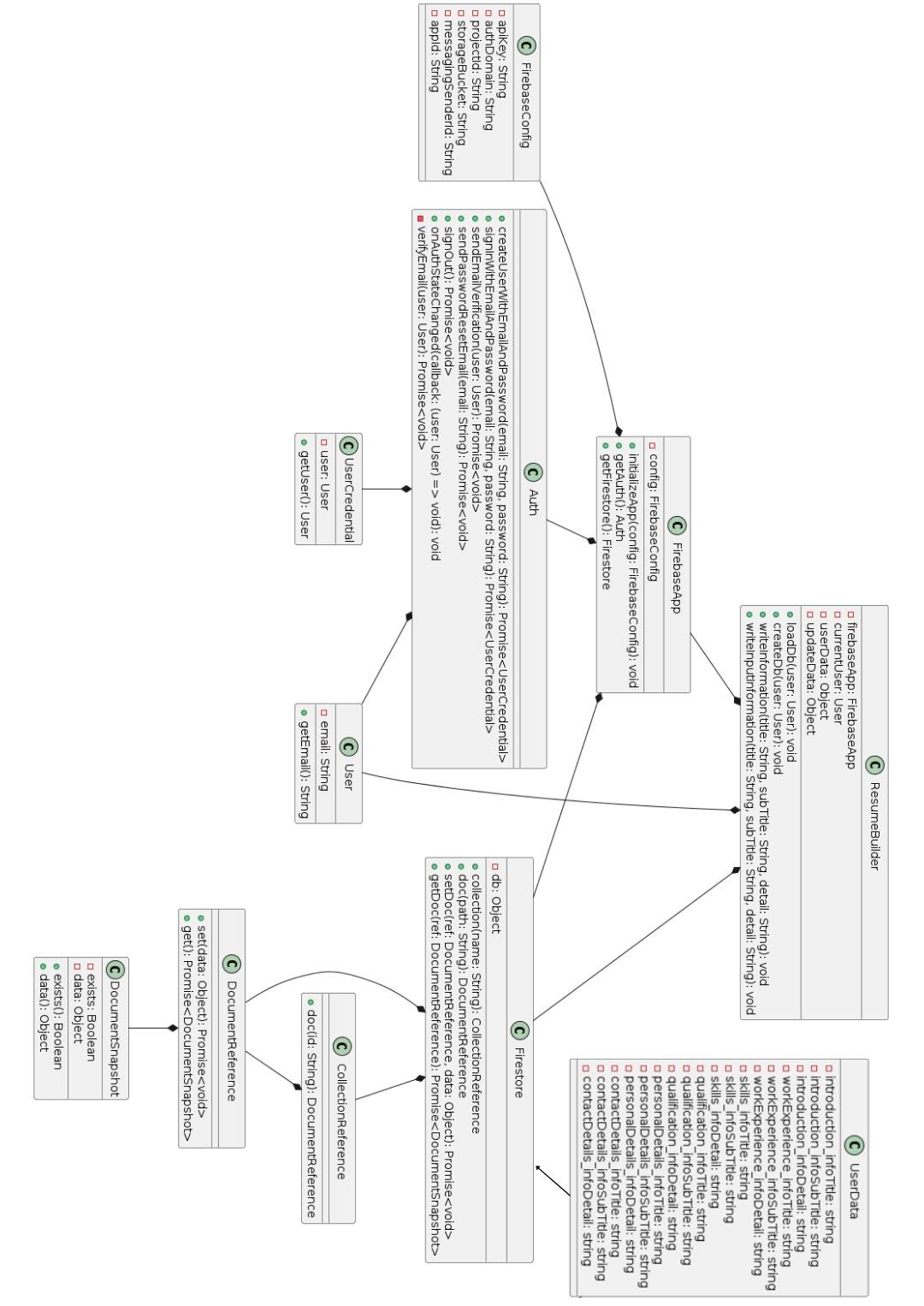
This sequence diagram offers a structured representation of the dynamic interactions between users and the StandOut 3D Resume Builder application during resume-related processes. By illustrating the step-by-step flow of actions and

dependencies, it provides valuable insights into the user journey and system functionality, enhancing comprehension and facilitating efficient system design and implementation.

#### CLASS DIAGRAM:

****

##### It's important to note that this diagram excludes the THREE.js 3D rendering aspect to maintain clarity and focus on the main functionalities.

****

Class Diagram Overview:

The Class Diagram presented here offers a structured representation of the StandOut 3D Resume Builder application's static view, showcasing the classes, interfaces, attributes, and methods that comprise the system. This diagram serves as a fundamental tool for modelling object-oriented systems, providing insights into the system's structure and the interactions between its components. Below is an expanded explanation of each class and its components:

FirebaseApp:

* + - Represents the Firebase application configuration.
    - Attributes:
      * config: An instance of FirebaseConfig containing configuration details.
    - Methods:
      * createApp(config: FirebaseConfig): Creates a Firebase app with the specified configuration.
      * getDefault(): Retrieves the default authentication instance.
      * getFirestore(): Retrieves the Firestore database instance.

Auth:

* + - Handles user authentication and authorization.
    - Attributes:
      * apiKey, authDomain, projectId, storageBucket, messagingSenderId, appId: Firebase authentication- related properties.
    - Methods:
      * createUserWithEmailAndPassword(email: String, password: String): Creates a new user account.
      * sendPasswordResetEmail(email: String): Sends a password reset email to the user.
      * sendEmailVerification(user: User): Sends an email verification link to the user.
      * signInWithEmailAndPassword(email: String, password: String): Authenticates the user with email and password.
      * signOut(): Logs the user out.
      * onAuthStateChanged(callback: (user: User) => void): Registers a callback for changes in user authentication state.

UserCredential:

* + - Represents the result of an authentication operation.
    - Attributes:
      * user: An instance of User representing the authenticated user.
    - Methods:
      * getUser(): Retrieves the associated user.

User:

* + - Represents a user account.
    - Attributes:
      * email: The user’s email address.
    - Methods:
      * getEmail(): Retrieves the user’s email.

ResumeBuilder:

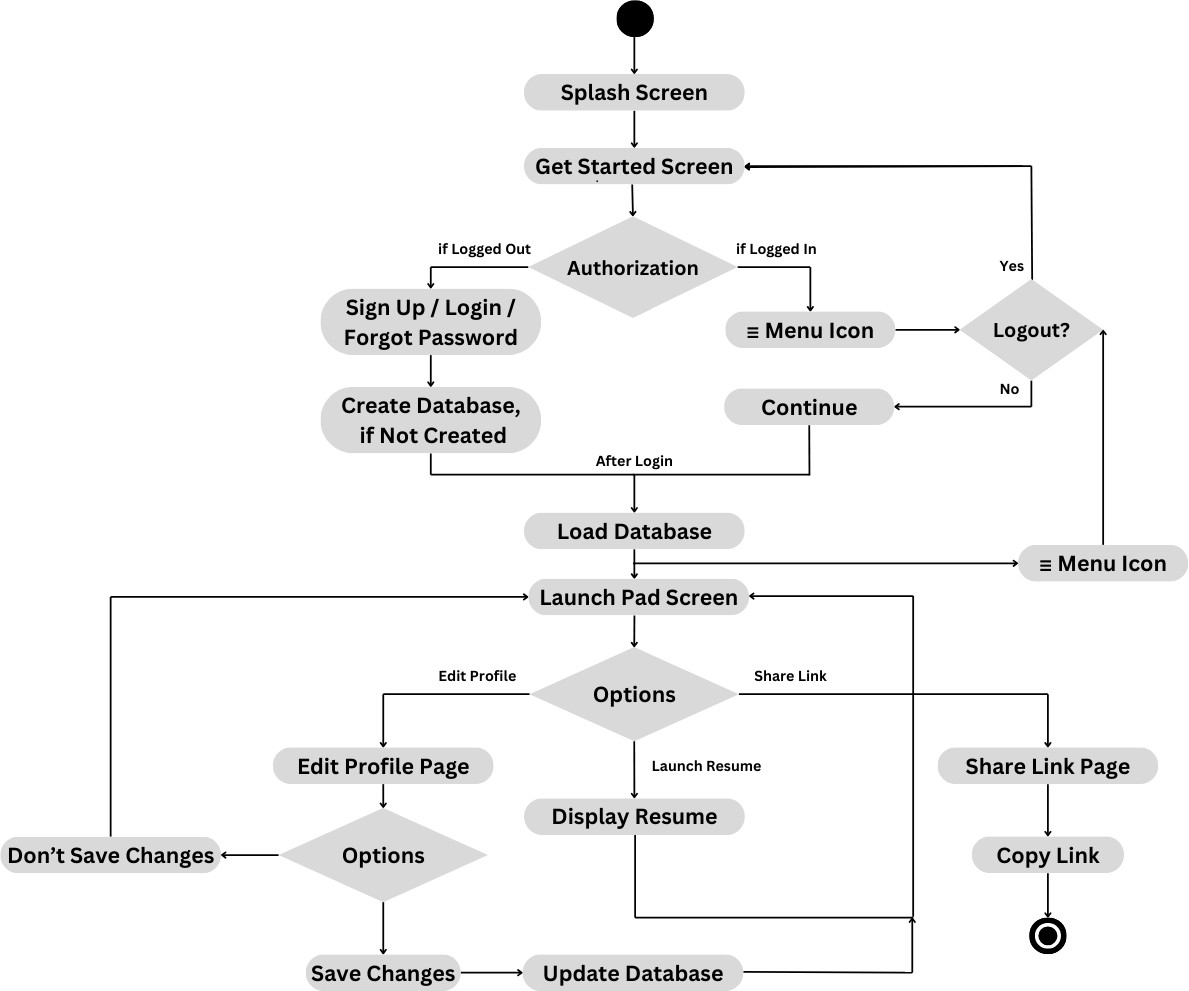
* + - Manages resume-related functionality.
    - Associations:
      * userManager: IUserManager: A reference to the user manager.
    - Methods:
      * updateData(data: Object): Updates resume data.
      * loadData(): Object: Loads resume data.
      * toDocument(user: IUser, void): Converts resume data to a document.
      * writeInformation(title: String, value: String, detail: String): Writes resume information.
      * returnToMainMenu(): Navigates back to the main menu.

UserData:

* + - Represents resume data attributes.
    - Attributes:
      * Various resume-specific attributes such as introduction, education, skills, work experience, languages, personal details, and contact details.

This detailed Class Diagram provides a holistic understanding of the system's structure, relationships, and functionality, facilitating efficient system design, implementation, and maintenance within the StandOut 3D Resume Builder application.

#### ACTIVITY DIAGRAM:

****

Activity Diagram Overview:

The Activity Diagram provided offers a visual representation of the flow of actions and interactions within the StandOut 3D Resume Builder application's user interface.

It guides users through various screens and functionalities, facilitating a seamless user experience.

Below is a detailed explanation of the depicted activities and their sequences:

Detailed Explanation:

**Splash Screen:**

* Represents the initial screen displayed upon launching the application.

Get Started Screen:

* Users transition from the splash screen to the get started screen, initiating the user journey.

Authorization:

* If the user is Logged Out, they encounter three options:
  + **Sign Up:** Enables users to create a new account.
  + **Login:** Allows users to authenticate with their credentials.
  + **Forgot Password:** Assists users in recovering their password.
* If the user is Logged In, they bypass authentication and directly access the Menu Icon.

Create Database (if Not Created):

* Ensures the setup of the database if it doesn't already exist.

Load Database:

* Retrieves data from the database, ensuring access to stored information.

Launch Pad Screen:

* Represents the main interface offering users three options:
  + **Edit Profile:** Navigates users to the "Edit Profile Page," where they can make changes to their resume.
  + **Save Changes:** Updates the database with any modifications made.
  + **Don’t Save Changes:** Allows users to discard changes if desired.
  + **Options:** Displays the resume for review.
  + **Share Link:** Provides users with a link for sharing their resume with others.

Arrows and decision points within the diagram signify the flow of actions and choices available to users throughout their interaction with the application.

It's important to note that this activity diagram represents the high-level flow, and each step may involve more detailed interactions and sub-processes within the application, ensuring a comprehensive user experience from login to resume sharing.

# DATA DICTIONARY

**DATA DICTIONARY**

|  |  |  |  |
| --- | --- | --- | --- |
| **NAME** | **NULL / NOTNULL** | **TYPE** | **KEY** |
| email\_id | Not Null | email | Primary |
| contactDetails\_infoDetail | Not Null | string |  |
| contactDetails\_infoSubTitle | Not Null | string |  |
| contactDetails\_infoTitle | Not Null | string |  |
| introduction\_infoDetail | Not Null | string |  |
| introduction\_infoSubTitle | Not Null | string |  |
| introduction\_infoTitle | Not Null | string |  |
| personalDetails\_infoDetail | Not Null | string |  |
| personalDetails\_infoSubTitle | Not Null | string |  |
| personalDetails\_infoTitle | Not Null | string |  |
| qualification\_infoDetail | Not Null | string |  |
| qualification\_infoSubTitle | Not Null | string |  |

|  |  |  |  |
| --- | --- | --- | --- |
| qualification\_infoTitle | Not Null | string |  |
| skills\_infoDetail | Not Null | string |  |
| skills\_infoSubTitle | Not Null | string |  |
| skills\_infoTitle | Not Null | string |  |
| workExperience\_infoDetail | Not Null | string |  |
| workExperience\_infoSubTitle | Not Null | string |  |
| workExperience\_infoTitle | Not Null | string |  |

# IMPLEMENTATION OF

**PROJECT**

**OVERVIEW OF TECHNOLOGIES**

**USED**

Front End Technologies

In the development of StandOut 3D Resume Builder, a robust arsenal of frontend technologies was employed to create a seamless and visually captivating user experience. Here's an in- depth exploration of the frontend technologies utilized**:**

**HTML, CSS, and JavaScript:** HTML (Hypertext Markup Language), CSS (Cascading Style Sheets), and JavaScript form the backbone of frontend development, providing the foundation for creating and styling web pages. HTML defines the structure of web content, while CSS controls its presentation and layout. JavaScript, on the other hand, adds interactivity and dynamic behavior to web pages, enabling features such as animations, form validation, and DOM manipulation.

In StandOut 3D Resume Builder, HTML, CSS, and JavaScript work in tandem to create the user interface elements, layout, and interactivity. HTML is used to structure the content of the resume builder interface, CSS styles the elements to enhance visual appeal and usability, and JavaScript adds interactive

features such as form validation, dynamic content updates, and user interactions.

**THREE.js:** THREE.js is a powerful JavaScript library used for creating and rendering 3D graphics in web browsers. It provides a wide range of features and functionalities for building immersive 3D environments, including support for rendering geometric shapes, textures, materials, lighting, and animations.

In the context of StandOut 3D Resume Builder, THREE.js plays a crucial role in creating the immersive 3D background environment that serves as the backdrop for the resume- building process. By leveraging THREE.js, users are transported into a virtual space filled with stars, planets, and other celestial objects, enhancing the overall user experience and engagement.

**Exploration of THREE.js Features:** THREE.js offers a plethora of features and functionalities that enable developers to create stunning 3D graphics and animations. Some of the key features explored in StandOut 3D Resume Builder include:

* **Rendering of 3D objects**: THREE.js provides APIs for creating and rendering 3D objects such as cubes, spheres, planes, and more.
* **Textures and materials:** Developers can apply textures and materials to 3D objects to give them realistic appearances and properties.
* **Lighting and shadows:** THREE.js supports various types of lighting, including ambient, directional, point, and spotlights, as well as shadows for added realism.
* **Animations:** THREE.js allows for the creation of dynamic animations using keyframe animations, morph targets, and skeletal animations.
* **Camera controls:** Users can navigate the 3D environment using intuitive camera controls, including pan, zoom, and orbit functionalities.

**Examples and Demonstrations:** To showcase the capabilities of THREE.js in StandOut 3D Resume Builder, several examples and demonstrations were created to illustrate its usage in various aspects of the project. These examples include:

* **Dynamic background animation**: Users can experience a visually stunning 3D background animation featuring moving stars, rotating planets, and other celestial objects.
* **Interactive elements:** THREE.js enables the creation of interactive elements within the 3D environment, allowing users to interact with objects and navigate through the scene.
* **Customization options:** Users can customize the appearance and behavior of the 3D environment using intuitive controls and settings, such as changing the background color, adjusting the lighting, and selecting different textures and materials.

**Integration with HTML, CSS, and JavaScript:** In StandOut 3D Resume Builder, THREE.js seamlessly integrates with HTML, CSS, and JavaScript to create a cohesive and immersive user experience. The 3D environment is embedded within the web page alongside other frontend elements, allowing for seamless interaction and navigation. JavaScript is used to control the behavior of the 3D scene, such as handling user inputs, updating object properties, and triggering animations.

**Conclusion:** In conclusion, the frontend technologies used in StandOut 3D Resume Builder play a crucial role in creating an immersive and engaging user experience. By harnessing the power of HTML, CSS, JavaScript, and THREE.js, the project offers users a unique platform to showcase their skills and experiences in a visually captivating 3D environment. Moving forward, these frontend technologies will continue to drive innovation and shape the future of web development, paving the way for more immersive and interactive experiences on the web.

**Background and Motivation:** The inspiration behind StandOut 3D Resume Builder stemmed from the realization that traditional resumes often fail to capture the attention of recruiters amidst a sea of applicants. Conventional paper or digital resumes lack the depth and interactivity needed to truly convey a candidate's unique value proposition. Motivated by this observation, the project aims to break free from the constraints of static resumes and introduce a dynamic and immersive experience for both job seekers and employers.

**Objectives and Goals:** The primary objective of StandOut 3D Resume Builder is to empower job seekers with a platform that goes beyond the limitations of traditional resumes. By leveraging cutting-edge technologies and innovative design principles, the project seeks to:

* Provide users with the tools to create visually stunning and interactive 3D resumes.
* Enhance user engagement and retention through immersive 3D backgrounds and dynamic content.
* Streamline the resume-building process while offering comprehensive customization options.
* Modernize the job application experience by embracing the potential of 3D visualization and interactivity.

**Importance of Distinctive Platform:** In today's digital age, where attention spans are short and competition is fierce, it is essential for job seekers to differentiate themselves from the crowd. StandOut 3D Resume Builder recognizes the importance of creating a distinctive platform that not only grabs the attention of recruiters but also leaves a lasting impression. By offering a unique and engaging user experience, the project aims to help job seekers break through the noise and stand out in a highly competitive job market.

**Significance of Modernizing Resume Building:** The traditional resume-building process has remained largely unchanged for decades, relying on static documents to convey a candidate's qualifications and experiences. However, in an increasingly digital and visually-driven world, there is a growing demand for innovative approaches to resume building. StandOut 3D Resume Builder seeks to address this need by harnessing the power of 3D graphics, interactivity, and immersive backgrounds to transform the way resumes are created, presented, and perceived. 22 30

# CODE

**HTML CODE:**

<!doctype html>

<html lang="en">

<head>

<meta charset="UTF-8" />

<link rel="icon" type="image/svg+xml" href="/src/img/SVG/ak-favicon.svg" />

<meta name="viewport" content="width=device-width, initial-scale=1.0" />

<title>StandOut</title>

</head>

<body style="background-color: black;">

<div

style="position: fixed !important; bottom: 0%

!important; width: 100% !important; height: 55px

!important; z-index: 9999 !important;">

<a href="https://tiiny.host?ref=free-site">

<div

style="padding: 10px !important; background: rgb(220,137,232) !important;

background: linear-gradient(141deg, rgba(220,137,232,1) 0%, rgba(144,64,204,1) 100%) !important; height: 55px; text-align: center">

<img style="height: 35px !important; width: auto

!important;" src="https://tiiny.host/assets/img/ad.png" alt="tiiny.host">

</div>

</a>

</div>

<canvas id="bg"></canvas>

<canvas id="bgg"></canvas>

<canvas id="bggg"></canvas>

<script type="module" src="/main.js"></script>

<div class="splash-screen-container" id="splashy">

<h1 class="splash-screen-heading">

<span class="logo-p1">Stand</span>

<span class="logo-p2">Out</span>

</h1>

</div>

<main>

<ul class="navbar navbar-off" id="navy">

<li class="logo"><img src="src\img\SVG\nosignin.svg" alt="">

<div class="logo-text"></div>

</li>

<li class="nav-title t1">RESUME</li>

<li class="nav-title t2"></li>

<!-- <li class="title t3"></li> -->

<li class="hamburger" id="hammy">&#9776;</div>

</ul>

<div class="info" id="infy">

<header id="info-title-container">

<h1 id="info-title"></h1>

<p id="info-sub-title"></p>

</header>

<section id="info-detail-container">

<ul id="info-detail" style="width: 80%;">

</ul>

</section>

</div>

</main>

<div class="no" id="splashy-outro">

<h1 class="splash-screen-outro- heading">◻◻◻◻◻◻◻◻</h1>

</div>

<div class="menu-display menu-off" id="menny">

<ul class="menu-option-list">

</ul>

</div>

<div class="navigate-info" id="navyInfy" style="display: none;">

Use Arrow Keys<br>OR<br>Use On-Screen Navigation Buttons <br>located at the edges of the screen

</div>

<div class="navigate-side left-bottom">&#9665;</div>

<div class="navigate-side right-up">&#9655;</div>

</body>

</html>

### Breaking Down The HTML

**Code**

Let's go through each part of the HTML code:

1. **Doctype Declaration**: **<!doctype html>** declares the document type and version of HTML being used.
2. HTML Document Structure:
   * **<html lang="en">**: Defines the root element of the HTML document with the language attribute set to English.
   * **<head>**: Contains meta-information about the HTML document, such as character encoding, viewport settings, and title.
   * **<meta charset="UTF-8" />**: Specifies the character encoding for the document as UTF-8.
   * **<link rel="icon" type="image/svg+xml" href="/src/img/SVG/ak-favicon.svg" />**: Links the favicon (website icon) file.
   * **<meta name="viewport" content="width=device- width, initial-scale=1.0" />**: Configures the viewport settings for responsive design.
   * **<title>StandOut</title>**: Sets the title of the HTML document.
3. Body Content:
   * **<body style="background-color: black;">**: Sets the background color of the body to black.
   * **<div>**: Contains a fixed-positioned bottom bar for displaying an advertisement link.
   * **<canvas>**: Creates three canvas elements with IDs **bg**, **bgg**, and **bggg**, likely used for background effects.
   * **<script type="module" src="/main.js"></script>**: Links an external JavaScript file (**main.js**) to the HTML document for scripting functionality.
   * **<div class="splash-screen-container" id="splashy">**: Container for the splash screen displayed before the main content loads.
   * **<main>**: Main content section of the document.
   * **<ul class="navbar navbar-off" id="navy">**: Defines a navigation bar with a logo, navigation titles, and a hamburger menu.
   * **<div class="info" id="infy">**: Container for displaying profile information.
   * **<div class="no" id="splashy-outro">**: Container for an outro message displayed after the splash screen.
   * **<div class="menu-display menu-off" id="menny">**: Container for displaying menu options, initially hidden.
   * **<div class="navigate-info" id="navyInfy" style="display: none;">**: Container for navigation instructions.
   * **<div class="navigate-side left- bottom">&#9665;</div>** and **<div class="navigate- side right-up">&#9655;</div>**: Navigation buttons for navigating between sections.

let's expand the explanation of the HTML code to include more details and cover various elements and functionalities.

1. **Watermark Trademark:**

The watermark trademark serves as a unique identifier for the website, ensuring that its ownership is clearly marked. In this HTML code, the watermark is represented by the logo or symbol placed strategically on the webpage. It serves to prevent unauthorized use of the website's content and design elements.

1. Background Canvas:

The HTML code includes three canvas elements: **bg**, **bgg**, and **bggg**, each serving a specific purpose in creating the desired background effect. These canvas elements act as layers where graphical elements or animations can be rendered dynamically. They play a crucial role in achieving the visual aesthetics and ambiance of the webpage.

1. Splash Screen:

The splash screen container, identified by the ID **splashy**, is responsible for displaying a captivating introductory screen when the webpage loads. It typically contains eye-catching graphics, animations, or messages that engage users and set the tone for their experience on the website. The content of the splash screen is often temporary and may transition to the main content of the webpage after a brief period.

1. Navbar and Logo:

The navbar, represented by the **navbar** class, contains essential navigation elements for the website. It typically includes links to different sections or pages of the website, facilitating user interaction and exploration. The logo, displayed within the navbar, serves as a visual representation of the website's brand identity, helping users identify and remember the site.

1. Info Section:

The **info** section, identified by the ID **infy**, is a critical part of the webpage where dynamic content is displayed. It often contains informational or descriptive text, images, or multimedia elements related to the main purpose or theme of the website. The content within this section may vary based on user interactions, data retrieval from external sources, or predefined templates.

1. Menu Display:

The menu display, identified by the ID **menny**, provides a user interface for accessing additional features, options, or settings of the website. It typically appears as an overlay or popup menu triggered by user actions, such as clicking a button or icon. The menu may contain lists of options, buttons, or links that allow users to navigate, customize, or interact with the website's functionality.

1. Navigation Controls:

The navigation controls, represented by the left and right arrow symbols (◁ and 𝖣), provide users with intuitive ways to navigate through the webpage's content. These controls are often positioned at the edges of the screen for easy access and are commonly used to browse through slideshows, galleries, or multi-page content. They enhance user experience by enabling seamless navigation without relying solely on traditional scrollbar or mouse interactions.

1. Splash Screen Outro:

The splash screen outro, identified by the class **splash-screen- outro-heading**, serves as a concluding element of the splash screen experience. It typically includes a farewell message or acknowledgment to users, thanking them for visiting the website or providing additional instructions for proceeding to the main content. The outro helps transition users from the introductory phase to the interactive phase of their website journey.

1. Script Inclusion:

The **<script>** tag with the **type="module"** attribute includes the JavaScript file **main.js**, which contains client-side code responsible for enhancing the interactivity and functionality of the webpage. JavaScript is used to manipulate DOM elements, handle user interactions, fetch data from external sources, and dynamically update the content or behavior of the webpage. It plays a crucial role in creating dynamic and responsive web experiences.

1. Responsive Design:

The HTML code incorporates responsive design principles to ensure optimal viewing and interaction experiences across various devices and screen sizes. Elements such as viewport meta tags (**<meta name="viewport">**), flexible layout

structures, and media queries are used to adapt the webpage's layout, typography, and styling to accommodate different screen resolutions, orientations, and input methods. This approach enhances usability and accessibility, allowing users to engage with the website seamlessly across desktops, laptops, tablets, and smartphones.

1. Dynamic Data Binding:

Throughout the HTML code, various elements are assigned unique IDs and classes to facilitate dynamic data binding and manipulation using JavaScript. These IDs and classes serve as hooks or targets for JavaScript functions to access, modify, or update specific content or behaviors at runtime. By establishing relationships between HTML elements and corresponding JavaScript functions or data sources, the webpage can respond dynamically to user input, external events, or backend data changes.

**JAVASCRIPT CODE:**

// Importing necessary modules and assets import './style.css';

import \* as THREE from 'three';

import \* as TWEEN from "@tweenjs/tween.js";

import { earthTexture, jupiterTexture, marsTexture, mercuryTexture, neptuneTexture, plutoTexture, saturnRingTexture, saturnTexture, starsTexture, sunTexture, uranusRingTexture, uranusTexture, venusTexture,

|  |  |  |  |
| --- | --- | --- | --- |
|  | get, | set, go, now |  |
| } | from | './src/img'; // | Adjust the path as needed |

import { initializeApp } from 'firebase/app' import { getAuth, createUserWithEmailAndPassword, sendEmailVerification, signInWithEmailAndPassword, sendPasswordResetEmail } from 'firebase/auth' import {

getFirestore, collection, getDocs, addDoc, doc, setDoc, getDoc,

} from 'firebase/firestore' const firebaseConfig = {

apiKey: " Confidential ", authDomain: "Confidential", projectId: " Confidential ", storageBucket: " Confidential ", messagingSenderId: " Confidential ", appId: " Confidential "

};

initializeApp(firebaseConfig) const db = getFirestore()

const auth = getAuth() let userData = {

};

let updateData = {

};

let currentUser;

const loadDb = (user) => {

const collectionName = "profile-database"; // Replace with your collection name

const documentId = user.email; // Replace with the actual document ID

const docRef = doc(collection(db, collectionName), documentId);

getDoc(docRef)

.then((doc) => {

if (doc.exists) { userData = doc.data();

if (userData === undefined) createDb(user);

// You can now use userData for further processing

} else {

// handle the case where the document doesn't

exist

}

})

createDb(user);

.catch((error) => {

});

}

const createDb = (user) => {

const collectionName = "profile-database"; // Replace with your collection name

const documentId = user.email; // Replace with your desired ID

const documentRef = doc(collection(db, collectionName), documentId);

setDoc(documentRef, sampleUserData)

.then(() => {

})

.catch((error) => {

});

}

const sampleUserData = {

introduction\_infoTitle: "Your Name Here", introduction\_infoSubTitle: "Your Subtitle Here", introduction\_infoDetail: `Replace this text with your

introduction details.`,

workExperience\_infoTitle: "Work Experience Section", workExperience\_infoSubTitle: `Your Subtitle Here`, workExperience\_infoDetail: `Replace this text with

your work experience details.`, skills\_infoTitle: "Skills Acquired", skills\_infoSubTitle: `Your Subtitle Here`,

skills\_infoDetail: `Replace this text with your skills details.`,

qualification\_infoTitle: "Qualification Acquired", qualification\_infoSubTitle: `Your Subtitle Here`, qualification\_infoDetail: `Replace this text with your

qualification details.`, personalDetails\_infoTitle: "Personal Details",

personalDetails\_infoSubTitle: `Your Name Detail`, personalDetails\_infoDetail: `Replace this text with

your personal details.`, contactDetails\_infoTitle: "Contact Details",

contactDetails\_infoSubTitle: `Your Name Goes Here`, contactDetails\_infoDetail: `Replace this text with

your contact details.`

};

1. Importing necessary modules and assets:
   * The **import** keyword is used to import modules or assets from external sources into the current JavaScript file.
   * **import './style.css';** imports a CSS file named **style.css**, which likely contains stylesheets for the webpage.
   * **import \* as THREE from 'three';** imports the **THREE** module from the 'three' package, which is a popular JavaScript 3D library.
   * **import \* as TWEEN from "@tweenjs/tween.js";** imports the **TWEEN** module from the 'tween.js' package, which is used for creating smooth animations.
2. Importing textures:
   * Multiple texture assets are imported using ES6 destructuring from the './src/img' directory.
   * These textures likely represent images used for various visual elements in the 3D scene, such as planets, stars, and celestial bodies.
3. Importing Firebase modules and initializing Firebase:
   * The **initializeApp** function is called from the Firebase **app** module to initialize the Firebase application with the provided configuration.
   * The **firebaseConfig** object contains configuration settings such as **apiKey**, **authDomain**, **projectId**, etc., required for Firebase initialization.
   * **const db = getFirestore()** initializes Firestore, the Cloud Firestore database service, and stores a reference to it in the **db** variable.
4. Initializing Firebase authentication:
   * **const auth = getAuth()** initializes Firebase Authentication and stores a reference to it in the **auth** variable.
5. Defining user data and update data objects:
   * **let userData = {};** declares an empty object **userData**

to store user data retrieved from Firestore.

* + **let updateData = {};** declares an empty object

**updateData** to store data to be updated in Firestore.

1. Defining functions to interact with Firestore:
   * **loadDb** function loads user data from Firestore based on the user's email.
   * **createDb** function creates a new document in the Firestore collection if it doesn't exist.
2. Defining sample user data:
   * **sampleUserData** is an object containing sample data for different sections of a user's profile.
   * It includes sections such as introduction, work experience, skills, qualification, personal details, and contact details, each with sample information.

// Creating a WebGL renderer

const renderer = new THREE.WebGLRenderer({ antialias: true, alpha: true });

renderer.setSize(window.innerWidth, window.innerHeight);

document.getElementById('bg').parentNode.insertBefore(re nderer.domElement, document.getElementById('bg'));

// Creating a scene and camera const scene = new THREE.Scene();

const camera = new THREE.PerspectiveCamera( 45,

window.innerWidth / window.innerHeight, 0.1,

1000

);

// Adding ambient light to the scene

const ambientLight = new THREE.AmbientLight(0x333333); scene.add(ambientLight);

// Loading textures and creating celestial bodies const textureLoader = new THREE.TextureLoader(); const sunGeo = new THREE.SphereGeometry(16, 30, 30); const sunMat = new THREE.MeshBasicMaterial({ map:

textureLoader.load(sunTexture) });

const sun = new THREE.Mesh(sunGeo, sunMat); scene.add(sun);

// Function to create planets

const createPlanete = (size, texture, position, ring)

=> {

const geo = new THREE.SphereGeometry(size, 30, 30); const mat = new THREE.MeshStandardMaterial({ map:

textureLoader.load(texture) });

const mesh = new THREE.Mesh(geo, mat); const obj = new THREE.Object3D(); obj.add(mesh);

if (ring) {

const ringGeo = new THREE.RingGeometry( ring.innerRadius,

ring.outerRadius, 32);

const ringMat = new THREE.MeshBasicMaterial({ map: textureLoader.load(ring.texture), side: THREE.DoubleSide

});

const ringMesh = new THREE.Mesh(ringGeo, ringMat); obj.add(ringMesh);

ringMesh.position.x = position; ringMesh.rotation.x = -0.5 \* Math.PI;

}

scene.add(obj); mesh.position.x = position; return { mesh, obj }

};

const mercury = createPlanete(3.2, mercuryTexture, 28);

const venus = createPlanete(5.8, venusTexture, 44); const earth = createPlanete(6, earthTexture, 62); const mars = createPlanete(4, marsTexture, 78); const jupiter = createPlanete(12, jupiterTexture,

100);

const saturn = createPlanete(10, saturnTexture, 138, { innerRadius: 10,

outerRadius: 20,

texture: saturnRingTexture

});

const uranus = createPlanete(7, uranusTexture, 176, { innerRadius: 7,

outerRadius: 12,

texture: uranusRingTexture

});

const neptune = createPlanete(7, neptuneTexture, 200); const pluto = createPlanete(2.8, plutoTexture, 216);

// Adding point lights to the scene

const pointLight = new THREE.PointLight(0xFFFFFF, 7000, 30000);

scene.add(pointLight);

const pointLightCamera = new THREE.PointLight(0xFFFFFF, 500, 300);

scene.add(pointLightCamera);

// Generating stars in the background const stars = [];

const addStar = () => {

const geometry = new THREE.SphereGeometry(0.2, 24, 24);

const material = new THREE.MeshStandardMaterial({ color: 0xffffff });

const star = new THREE.Mesh(geometry, material); const [x, y, z] = Array(3).fill().map(() =>

THREE.MathUtils.randFloatSpread(10000000)); star.position.set(x, y, z); scene.add(star);

stars.push(star);

};

Array(700).fill().forEach(addStar);

// Function to animate stars function animateStars() {

for (const star of stars) {

star.scale.x = star.scale.y = star.scale.z = Math.sin(Date.now() \* 0.003 + stars.indexOf(star)) \*

0.35 + 1;

const maxPosition = 250;

if (Math.abs(star.position.x) > maxPosition || Math.abs(star.position.z) > maxPosition || Math.abs(star.position.z) > maxPosition) {

star.position.set(THREE.MathUtils.randFloatSpread(maxPos

ition), THREE.MathUtils.randFloatSpread(maxPosition), THREE.MathUtils.randFloatSpread(maxPosition));

}

}

}

// Setting initial animation parameters let current = 0;

let selfRotationSpeed = 1;

let aroundSunRotationSpeed = 0.4;

// Tween animation function

const tweenAnimate = (toPosX, toPosY, toPosZ, toRotX, toRotY, toRotZ) => {

let time = 3000

const tween = new TWEEN.Tween({ posX: camera.position.x, posY: camera.position.y, posZ: camera.position.z, rotX: camera.rotation.x, rotY: camera.rotation.y, rotZ: camera.rotation.z })

.to({ posX: toPosX, posY: toPosY, posZ: toPosZ, rotX: toRotX, rotY: toRotY, rotZ: toRotZ }, time)

.onUpdate((obj) => {

// Update camera position and rotation during the

tween

})

camera.position.set(obj.posX, obj.posY, obj.posZ); camera.rotation.set(obj.rotX, obj.rotY, obj.rotZ);

.easing(TWEEN.Easing.Exponential.Out) tween.start();

return tween;

};

// Creating a cube and a torus const cubeMaterials = [

new THREE.MeshStandardMaterial({ map: textureLoader.load(get) }),

new THREE.MeshStandardMaterial({ map: textureLoader.load(go) }),

new THREE.MeshStandardMaterial({ map: textureLoader.load(now) }),

new THREE.MeshStandardMaterial({ map: textureLoader.load(now) }),

new THREE.MeshStandardMaterial({ map: textureLoader.load(now) }),

new THREE.MeshStandardMaterial({ map: textureLoader.load(set) })

];

const cubeGeometry = new THREE.BoxGeometry(7.5, 7.5, 7.5);

const cube = new THREE.Mesh(cubeGeometry, cubeMaterials);

cube.position.set(-504.1526130620916, 19.975781533168185, -1039.4067039450292);

cube.rotation.set(-3.122376573243678, - 0.45153470030279186, -3.1332069401291682);

scene.add(cube);

const torusGeometry = new THREE.TorusGeometry(8.79, 2.35, 16, 100);

let torusHue = 1; // Initial hue value

const torusMaterial = new THREE.MeshStandardMaterial({ color: new THREE.Color().setHSL(torusHue, 1, 0.5),

});

const torus = new THREE.Mesh(torusGeometry, torusMaterial);

torus.position.set(-504.1526130620916, 19.975781533168185, -1039.4067039450292);

torus.rotation.set(-3.122376573243678, 1.1, -

3.1332069401291682);

scene.add(torus); pointLightCamera.intensity = 2250; let Hue = 6;

let's break down the provided code into detailed bullet points and paragraphs:

1. WebGL Renderer Setup:
   * A WebGL renderer is created with antialiasing and alpha transparency enabled.
   * The renderer's size is set to match the window's dimensions.
   * The renderer's DOM element is inserted into the HTML document before the element with the ID 'bg'.
2. Scene and Camera Initialization:
   * A new three.js scene and perspective camera are created.
   * The camera's parameters are set to define its field of view, aspect ratio, and clipping planes.
3. Ambient Light Addition:
   * An ambient light with a dark gray color (0x333333) is added to the scene.
   * Ambient light simulates indirect light that uniformly illuminates all objects in the scene.
4. Loading Textures and Creating Celestial Bodies:
   * Textures are loaded using the **TextureLoader** for various celestial bodies.
   * A function **createPlanete** is defined to create planets and other objects in the scene.
   * Each celestial body, such as the sun and planets, is created using a sphere geometry with a specified size and texture.
5. Point Light Addition:
   * Two point lights are added to the scene to simulate light sources.
   * **pointLight** is a powerful light source, while **pointLightCamera** is a weaker light source positioned near the camera.
6. Star Generation:
   * A function **addStar** is defined to create stars and add them to the scene.
   * 700 stars are generated and randomly positioned in space.
   * These stars are then added to the scene to create a starry background.
7. Star Animation Function:
   * A function **animateStars** is defined to animate the stars in the scene.
   * It updates the scale and position of each star over time, giving the appearance of twinkling stars.
   * Additionally, it ensures that stars remain within a certain range of positions.
8. Initial Animation Parameters:
   * Initial animation parameters are set to control the rotation speeds of celestial bodies.
   * **selfRotationSpeed** and **aroundSunRotationSpeed** variables control the rotational speeds of celestial bodies.
9. Tween Animation Function:
   * A function **tweenAnimate** is defined to animate the camera's position and rotation using tweening.
   * Tweening is a technique used to smoothly transition between different values over time.
10. Cube and Torus Creation:
    * A cube and a torus are created using predefined geometries and materials.
    * These objects are positioned and rotated in the scene and then added to the scene.
11. Hue Initialization:
    * A variable **Hue** is initialized with a value of 6. This variable may be used to control the color of the torus in the scene.

// Animation function const animate = () => {

changeWatermark() TWEEN.update();

pointLightCamera.position.set(camera.position.x, camera.position.y, camera.position.z);

animateStars(); sun.rotateY(selfRotationSpeed \* 0.001);

mercury.mesh.rotateY(selfRotationSpeed \* 0.004);

venus.mesh.rotateY(selfRotationSpeed \* 0.002);

earth.mesh.rotateY(selfRotationSpeed \* 0.02);

mars.mesh.rotateY(selfRotationSpeed \* 0.018);

jupiter.mesh.rotateY(selfRotationSpeed \* 0.04);

saturn.mesh.rotateY(selfRotationSpeed \* 0.038);

uranus.mesh.rotateY(selfRotationSpeed \* 0.03);

neptune.mesh.rotateY(selfRotationSpeed \* 0.032);

pluto.mesh.rotateY(selfRotationSpeed \* 0.008);

mercury.obj.rotateY(aroundSunRotationSpeed \* 0.03);

venus.obj.rotateY(aroundSunRotationSpeed \* 0.015);

earth.obj.rotateY(aroundSunRotationSpeed \* 0.01);

mars.obj.rotateY(aroundSunRotationSpeed \* 0.008);

jupiter.obj.rotateY(aroundSunRotationSpeed \* 0.002);

saturn.obj.rotateY(aroundSunRotationSpeed \* 0.0009);

uranus.obj.rotateY(aroundSunRotationSpeed \* 0.0004); neptune.obj.rotateY(aroundSunRotationSpeed \*

0.0001);

pluto.obj.rotateY(aroundSunRotationSpeed \* 0.00007); switch (current) {

case 0:

torus.rotation.y += 0.01;

cube.rotation.y -= 0.003;

torusHue += 0.001;

torusHue = (torusHue + 1) % 1; torus.material.color.setHSL(torusHue, 1, 0.5); break;

case 1:

torus.rotation.y -= 0.01;

cube.rotation.y += 0.006;

torusHue += 0.001;

torusHue = (torusHue + 1) % 1; torus.material.color.setHSL(torusHue, 1, 0.5);

break;

case 5:

torus.rotation.y += 0.01;

cube.rotation.y -= 0.003;

torusHue += 0.001;

torusHue = (torusHue + 1) % 1; torus.material.color.setHSL(torusHue, 1, 0.5); break;

case 6:

|  |  |  |
| --- | --- | --- |
| torus.rotation.y += 0.01; |  | |
| cube.rotation.y -= 0.003; |
| torusHue += 0.001; |
| torusHue = (torusHue + 1) % 1;  torus.material.color.setHSL(torusHue, | 1, | 0.5); |
| break; |  |  |

case 7:

// Torus Animations torus.rotation.y += 0.01;

cube.rotation.y = -0.6553470030279186;

torusHue += 0.001;

torusHue = (torusHue + 1) % 1; torus.material.color.setHSL(torusHue, 1,0.5); break;

case 17:

torus.rotation.y += 0.01;

cube.rotation.y -= 0.003;

torusHue += 0.001;

torusHue = (torusHue + 1) % 1;

torus.material.color.setHSL(torusHue, 1, 0.5); break;

default: break;

}

renderer.render(scene, camera)

}

1. Watermark Change Function:
   * The function **changeWatermark()** is called to handle any changes to the watermark or trademark on the webpage.
2. Tween Update:
   * The **TWEEN.update()** function is called to update any active tween animations in the scene.
3. Point Light Camera Position Update:
   * The position of the **pointLightCamera** is updated to match the position of the main camera.
4. Star Animation:
   * The function **animateStars()** is called to animate the stars in the background.
5. Celestial Bodies Rotation:
   * Each celestial body, including the sun, planets, and moons, is rotated according to its individual rotation speed (**selfRotationSpeed** and **aroundSunRotationSpeed**).
   * The rotation of each celestial body is calculated and applied based on its current rotation speed.
6. Cube and Torus Animation:
   * A **switch** statement is used to determine the current state of the animation and apply specific transformations accordingly.
   * Different cases within the switch statement correspond to different animation states.
   * For each case:
     + The rotation of the torus and cube is adjusted based on the current state.
     + The hue of the torus material is gradually shifted to create a color-changing effect.
7. Rendering:
   * After all animations and transformations are applied, the **renderer** renders the updated scene from the perspective of the camera.

**THE OVERALL VIEW OF THE**

**CODE:**

1. **Firebase Initialization**: You've properly initialized Firebase in your application using the provided configuration object **firebaseConfig**. This setup allows you to use Firebase services like Firestore and Authentication.
2. **Database Operations**: You've implemented functions to interact with Firestore, such as **loadDb** and **createDb**, to retrieve and store user data. This is essential for managing user profiles and information.
3. **3D Scene Setup**: Your Three.js setup includes creating a renderer, scene, camera, lights, and various celestial bodies like the sun and planets. You're also animating these objects to create a dynamic scene.
4. **UI and Interactivity**: You've added UI elements like buttons and navigation bars for user interaction. Functions like **changePage** and **handleShareEvent** demonstrate how users can navigate through different sections of the application.
5. **Animation and Effects**: You've incorporated animations for celestial bodies and background stars, adding visual

appeal to your application. Additionally, you're using tweening libraries like TWEEN.js for smooth transitions.

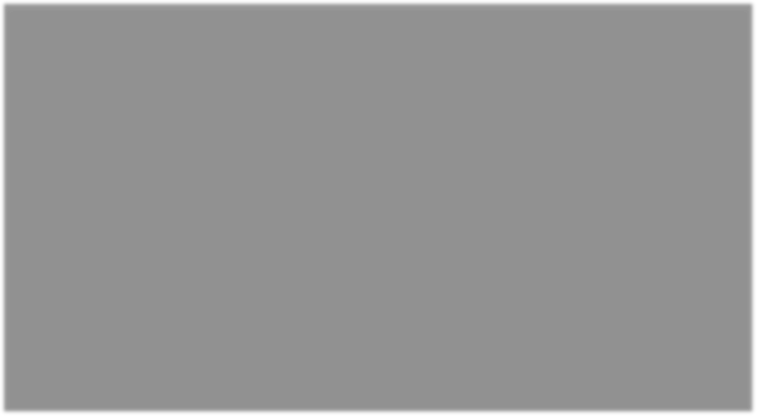
1. **Input Sanitization**: Your **sanitizeInput** function helps prevent XSS attacks by escaping HTML characters. This is crucial for maintaining security when dealing with user- generated content.
2. **Code Organization**: Consider organizing your code into modules or classes for better maintainability and readability. This can help manage complexity as your application grows.
3. **Error Handling**: Ensure robust error handling for Firebase operations and other asynchronous tasks. Proper error handling enhances the user experience and helps diagnose issues during development.
4. **Optimization**: Continuously optimize your code for performance, especially in animations and rendering. This includes minimizing unnecessary calculations and leveraging browser rendering capabilities.
5. **Documentation**: Document your code, especially complex logic and functions, to aid future maintenance and collaboration.

**IMPORTANT NOTE:**

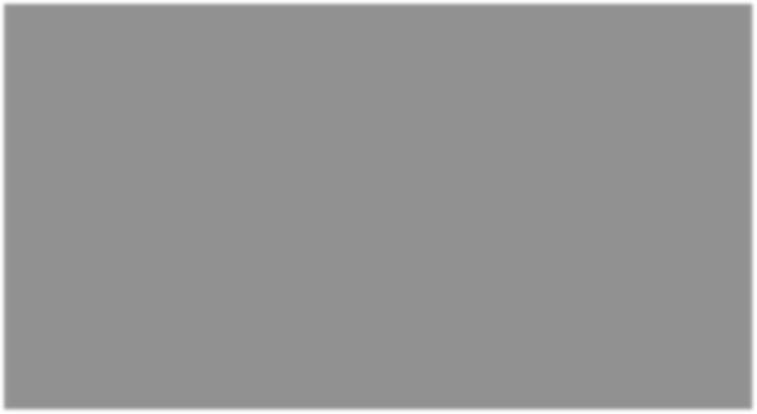
**The explanation provided above regarding the JavaScript code represents only a primary overview. The complete codebase of the project consists of over 1000 lines, making it impractical to cover every detail within this context. Therefore, only the essential aspects are addressed here. Should further clarification or details be necessary, it is advisable to directly consult the creator of the code.**

## SNAPSHOTS/OUTPUT DESIGNS:

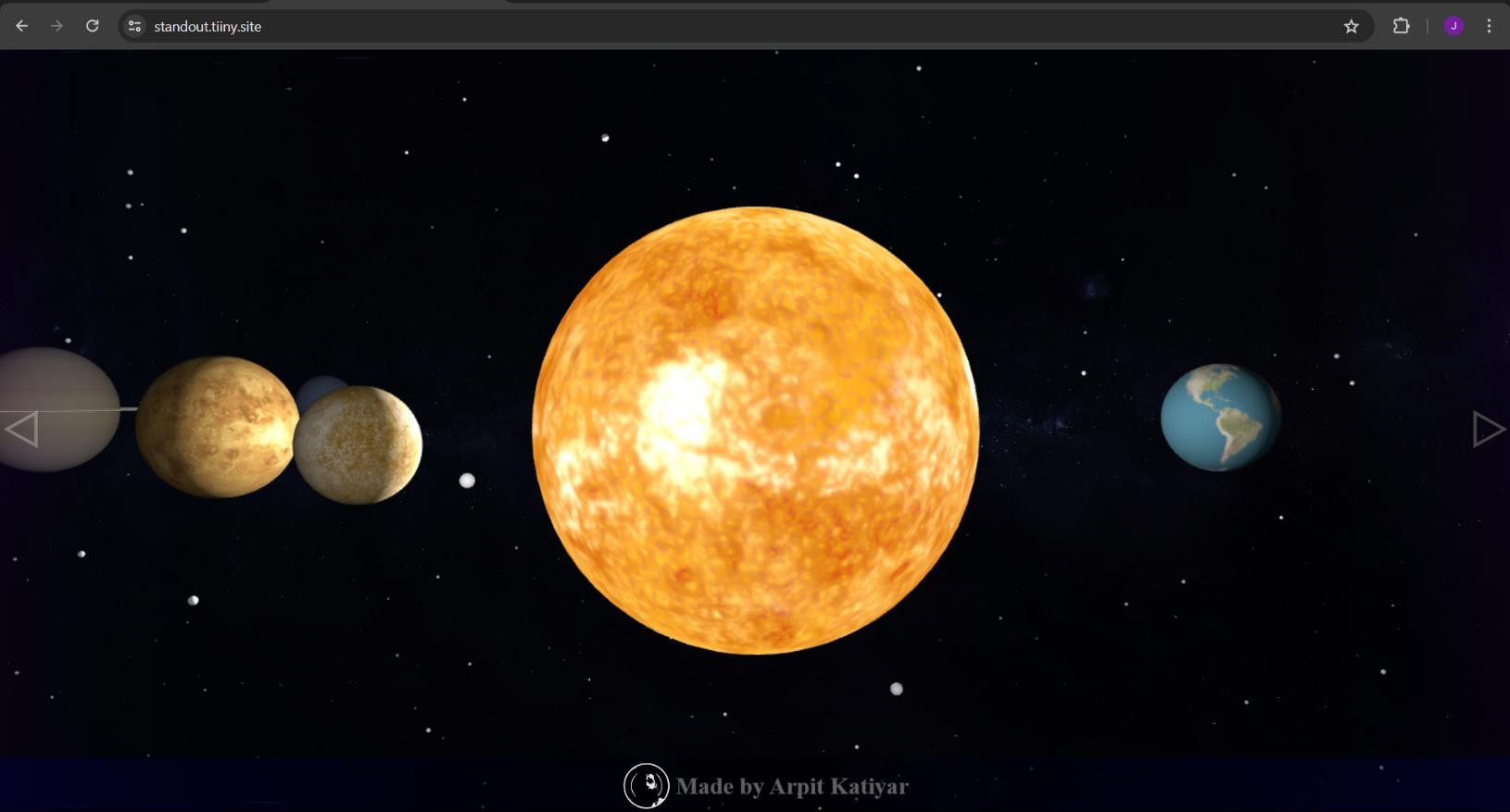
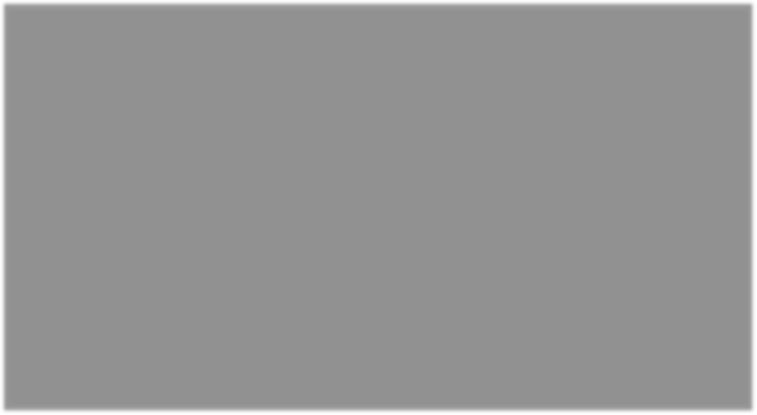
**SPLASH SCREEN:**



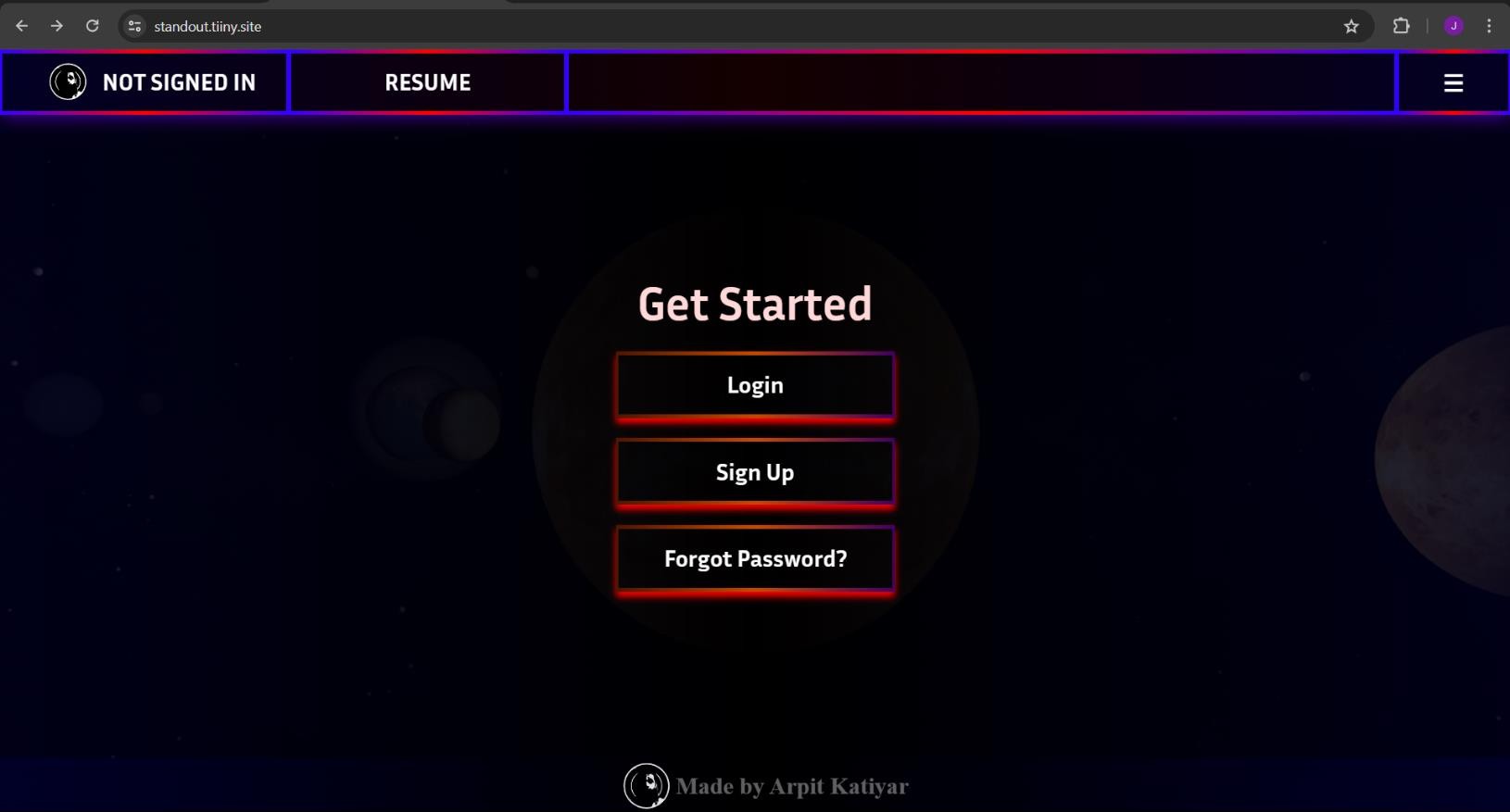
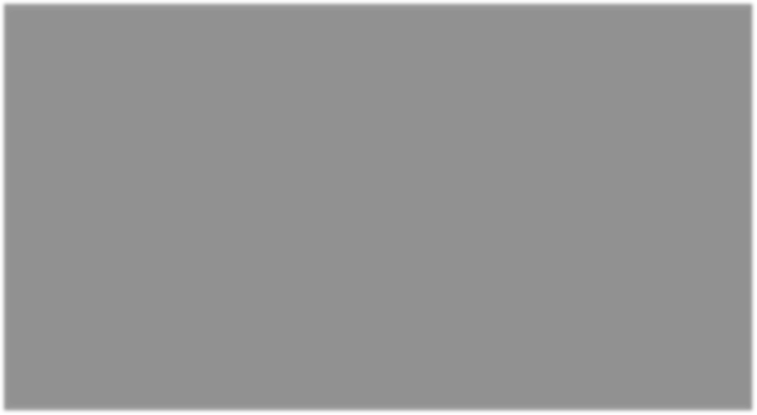
**NAVIGATION INSTRUCTION SCREEN:**



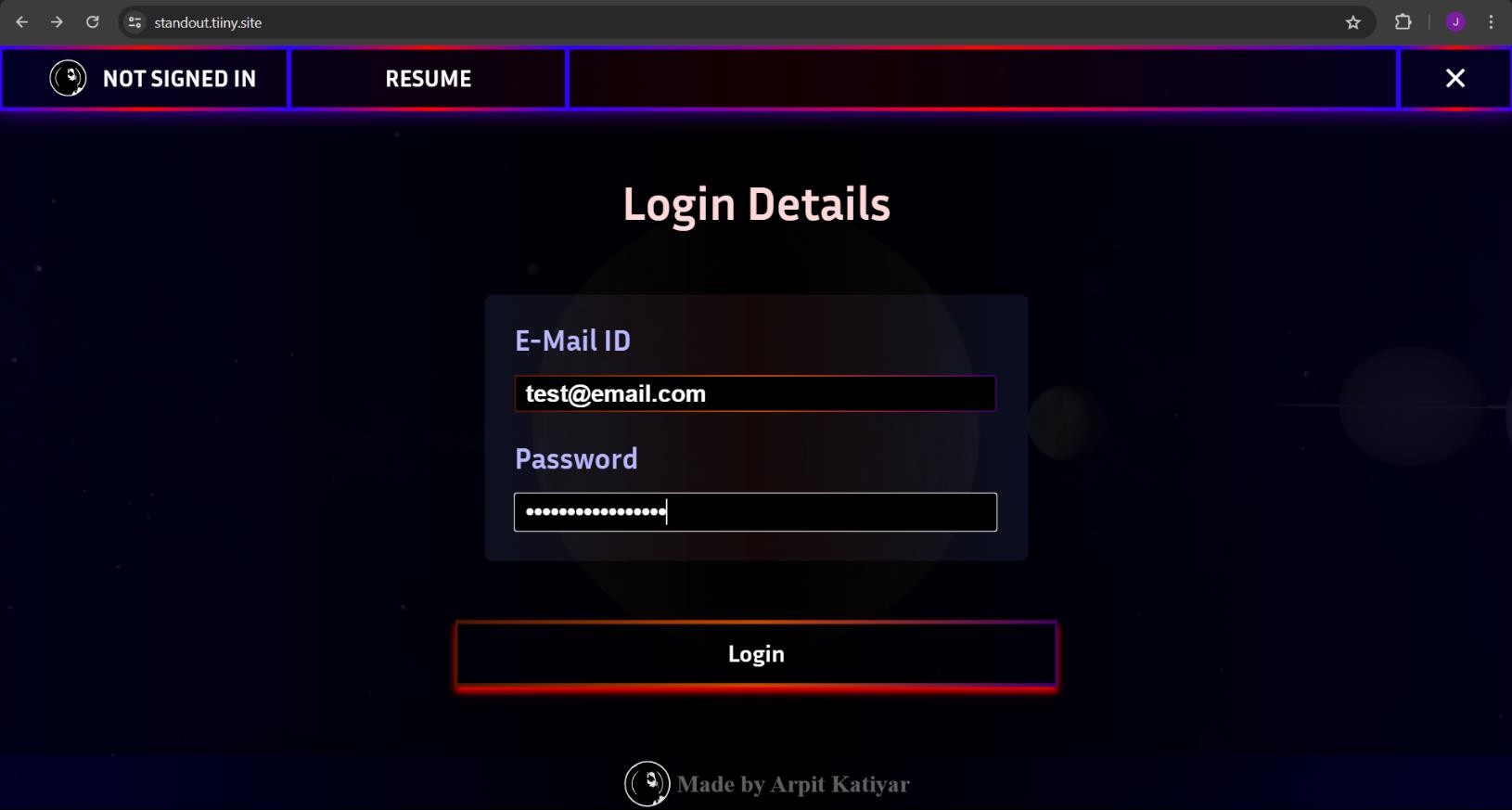
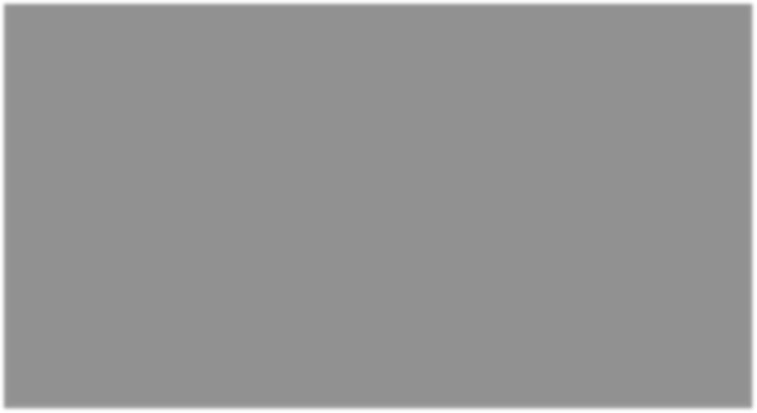
**BACKGROUND OVERVIEW:**



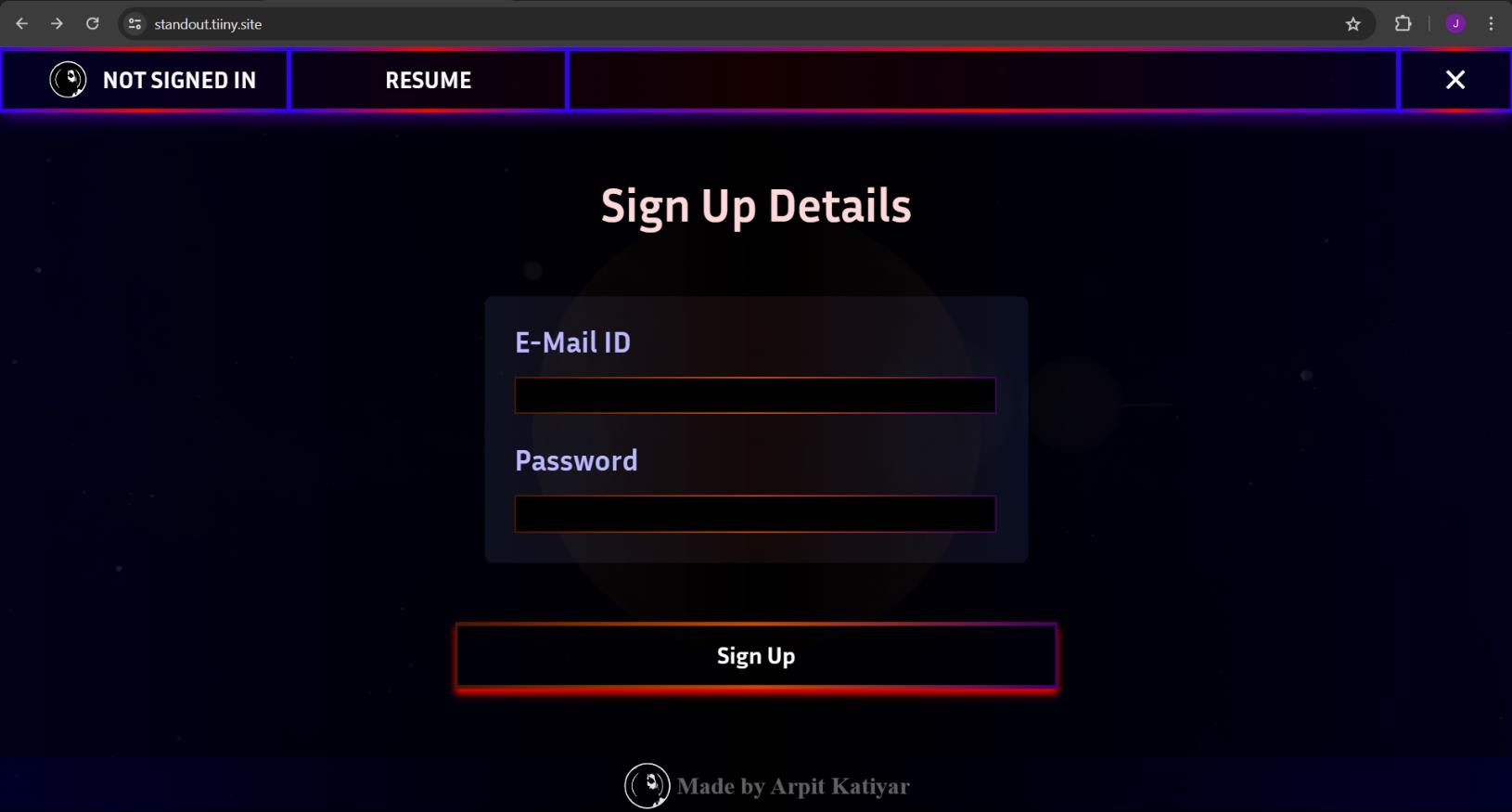
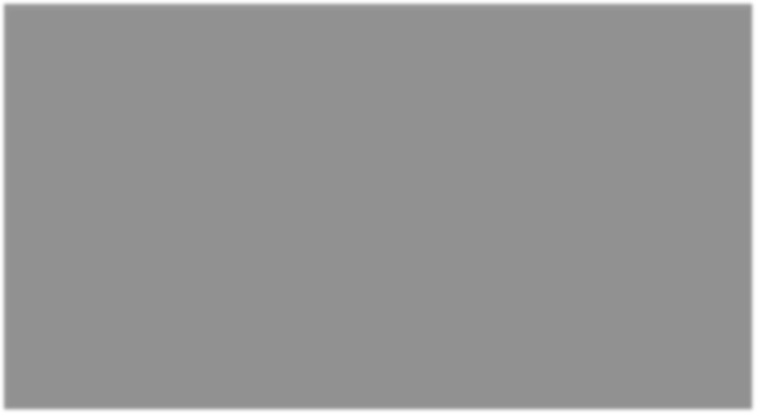
**GET STARTED SCREEN:**



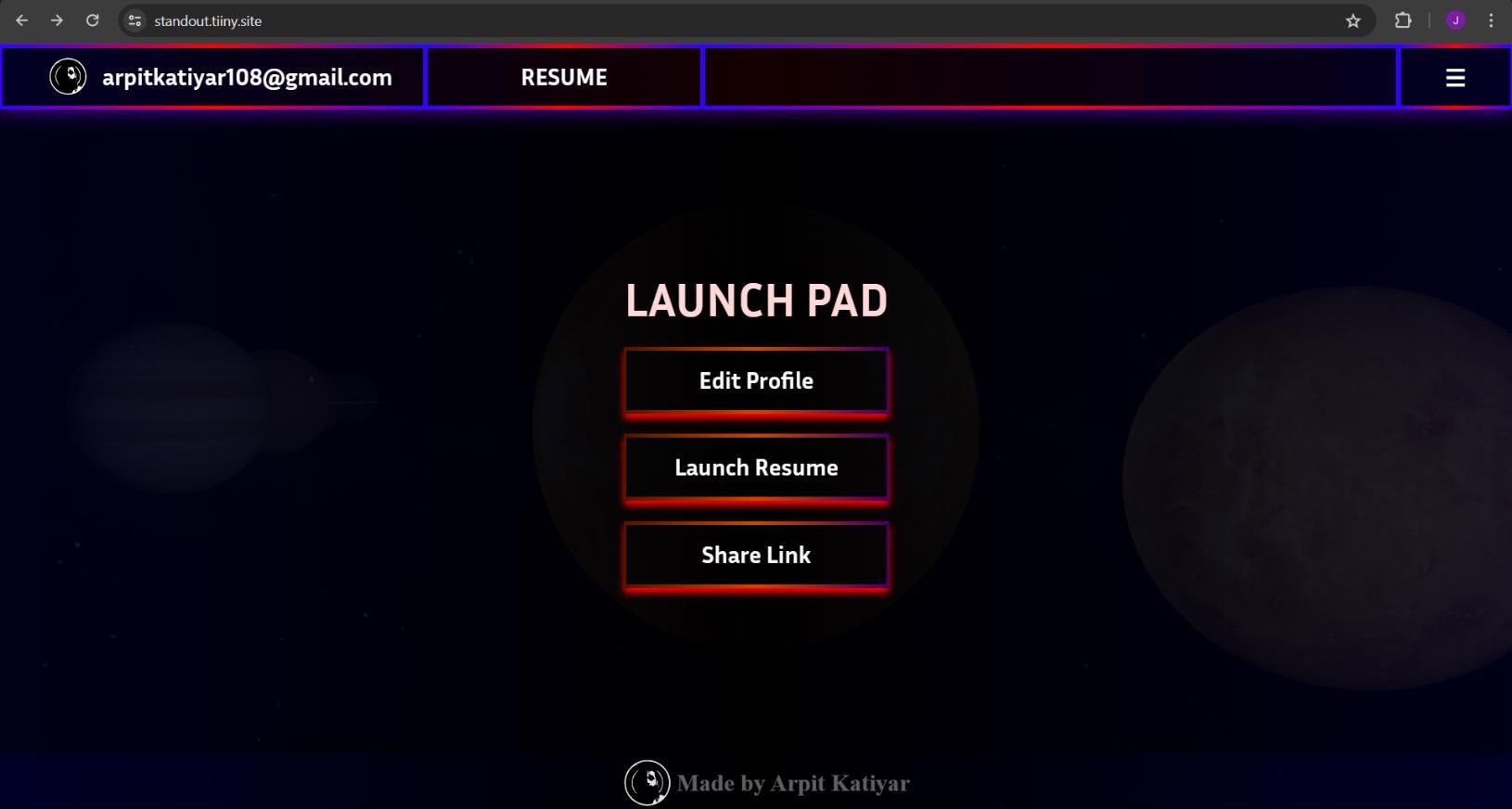
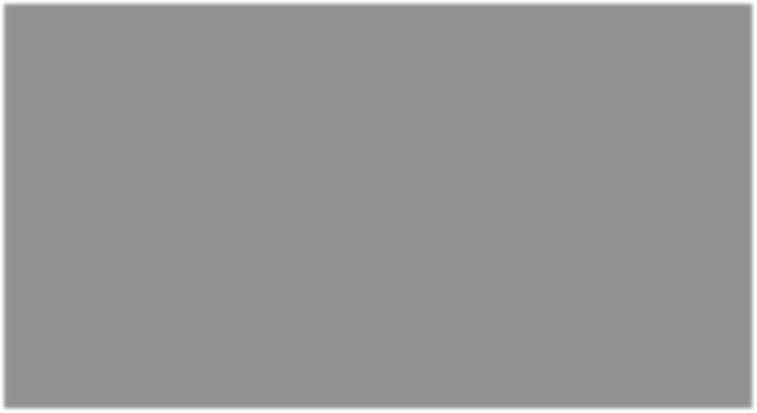
**LOG IN SCREEN:**



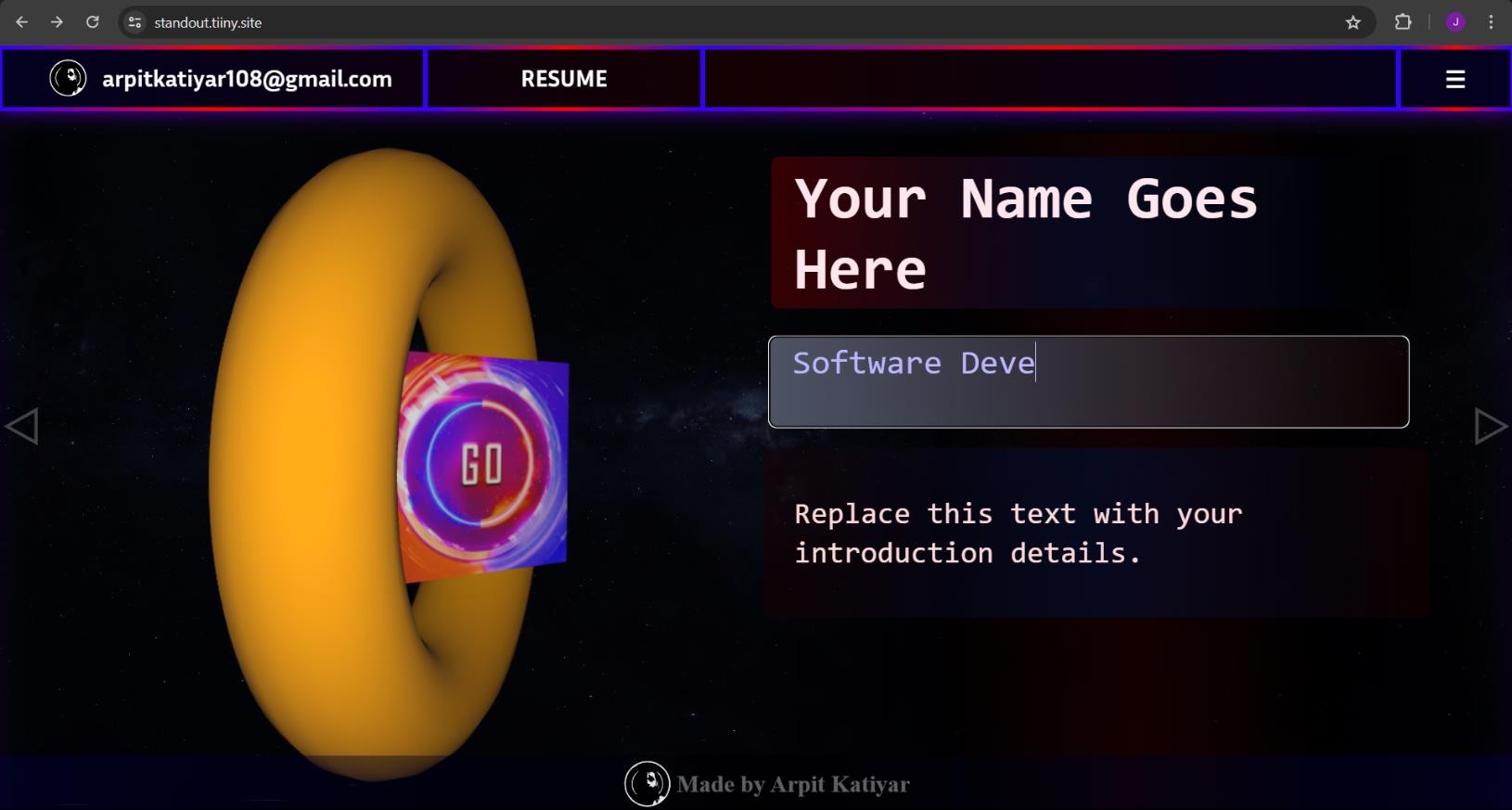
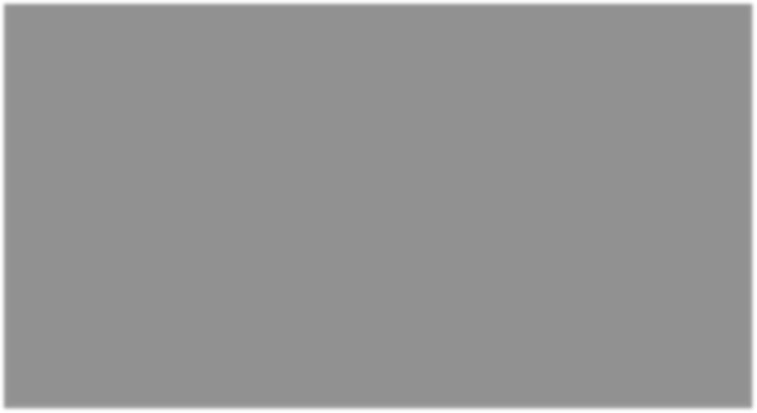
**SIGN UP SCREEN:**



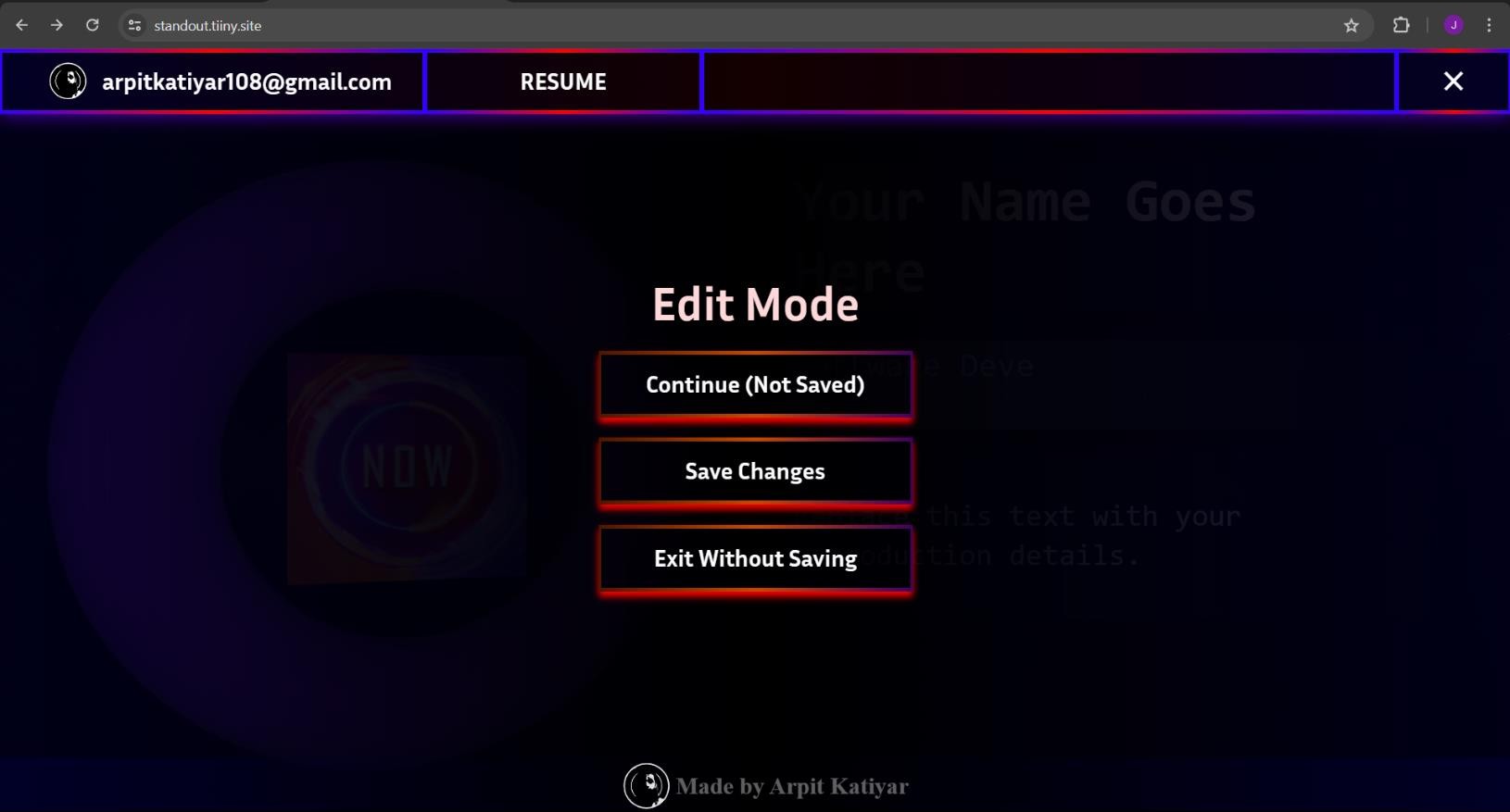
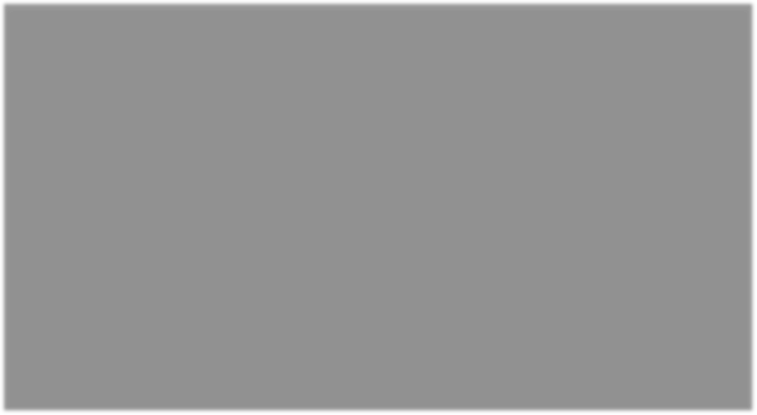
**LAUNCH PAD:**



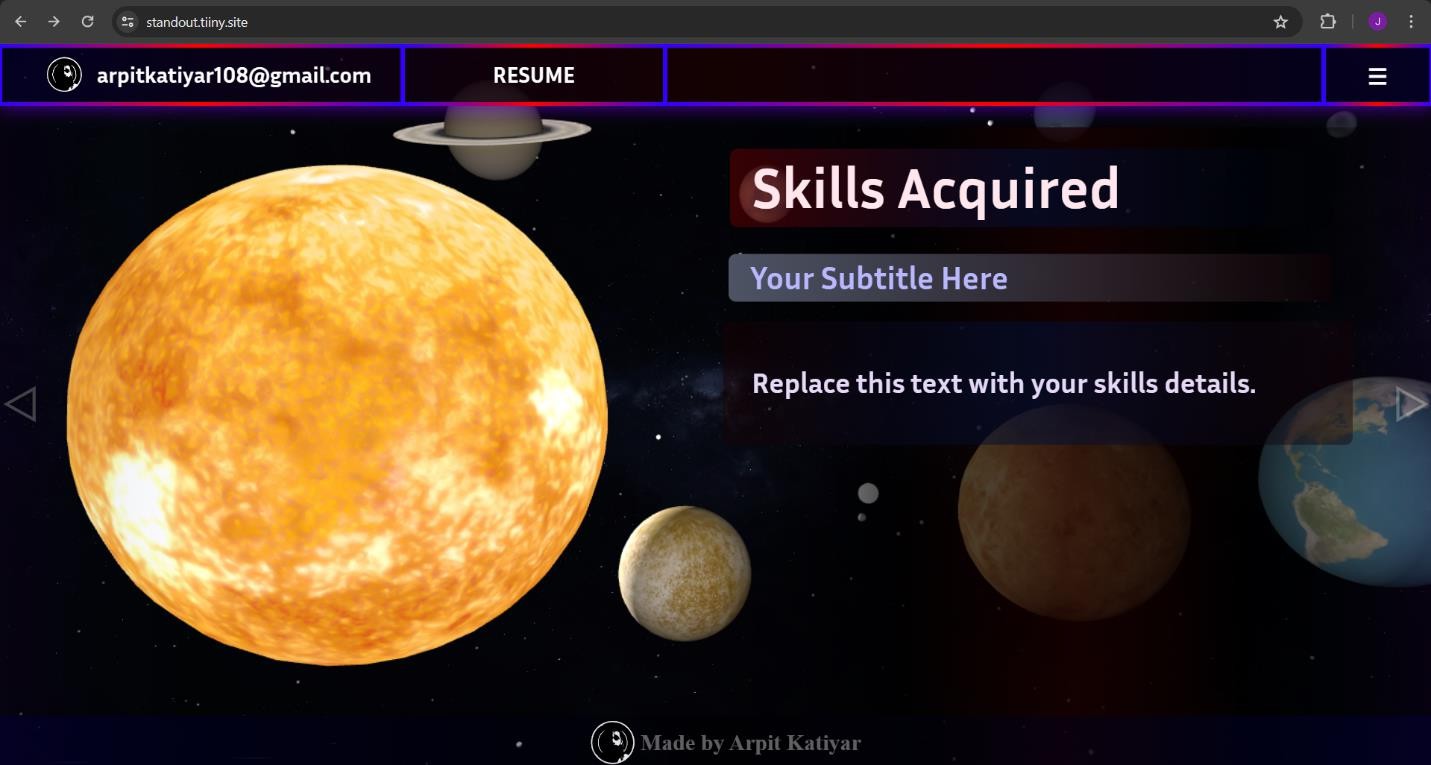
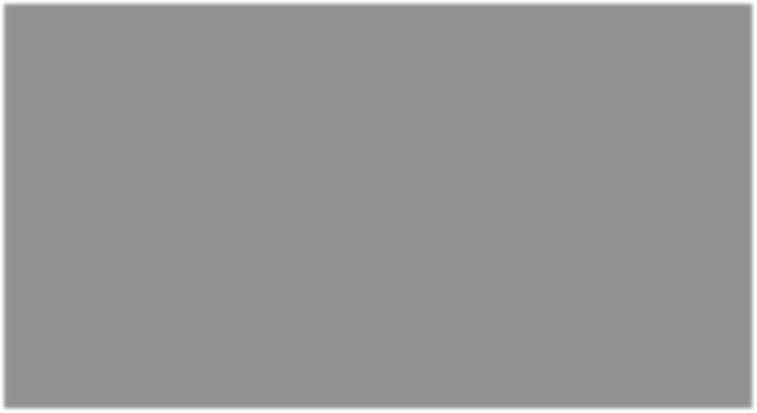
**EDIT PROFILE SCREEN:**



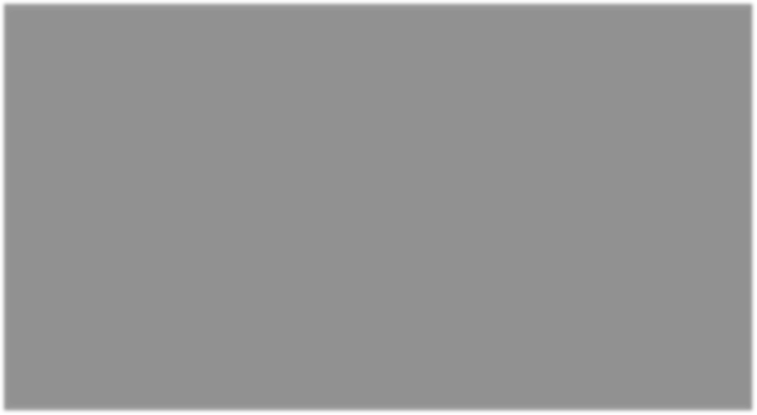
**EDIT MODE OPTIONS:**



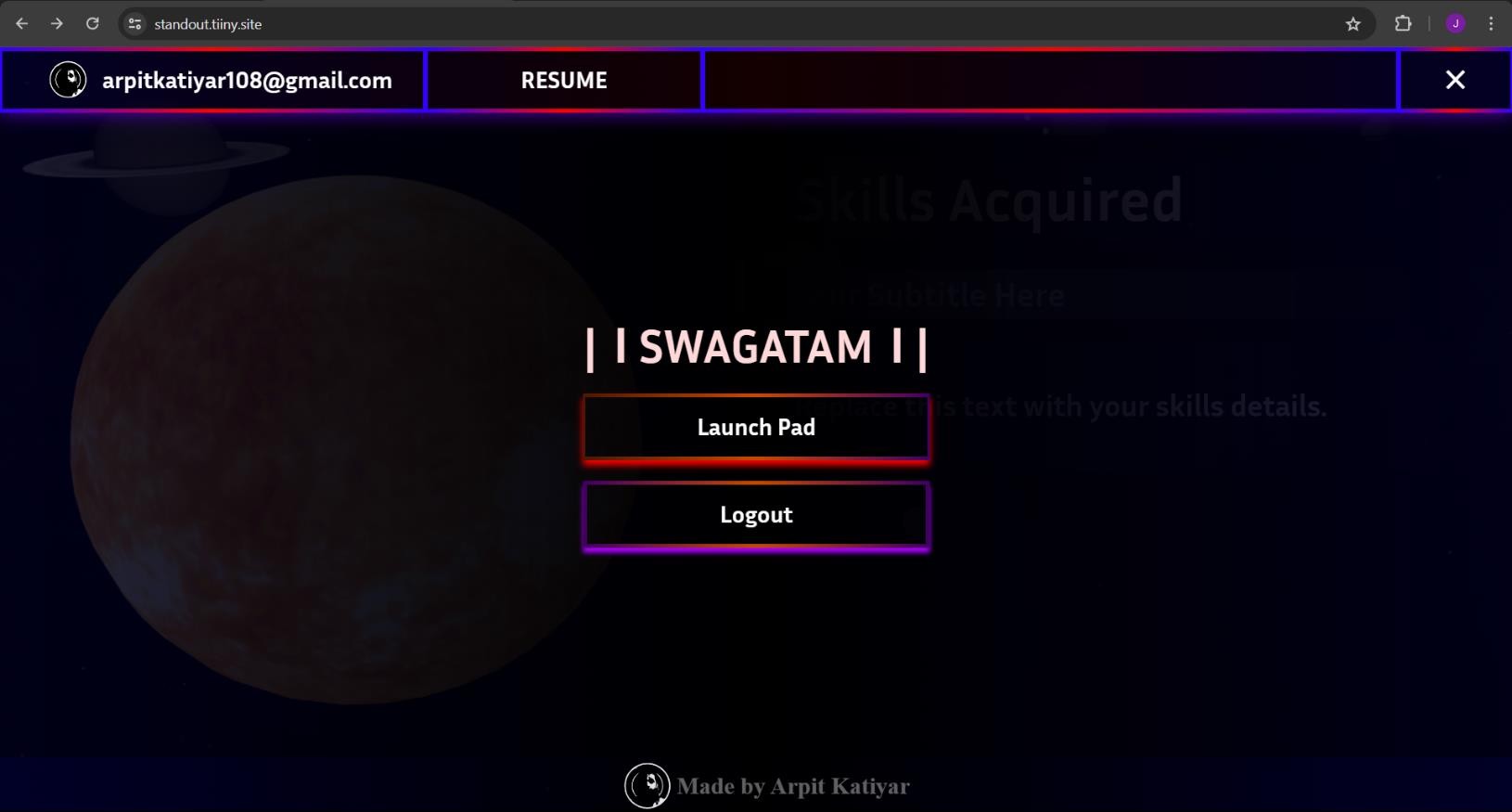
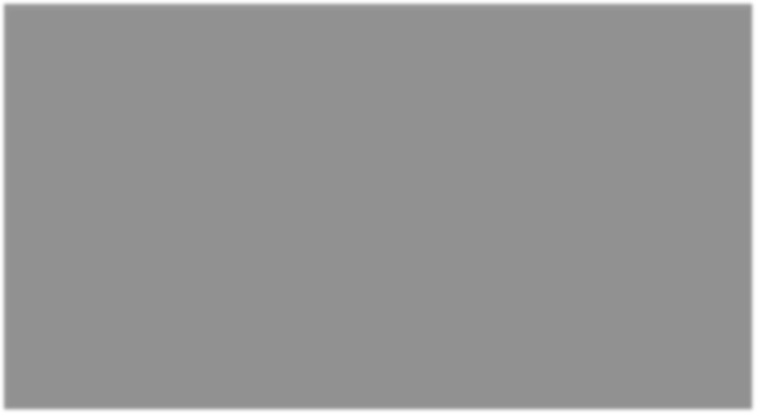
**LAUNCH PROFILE PAGES:**



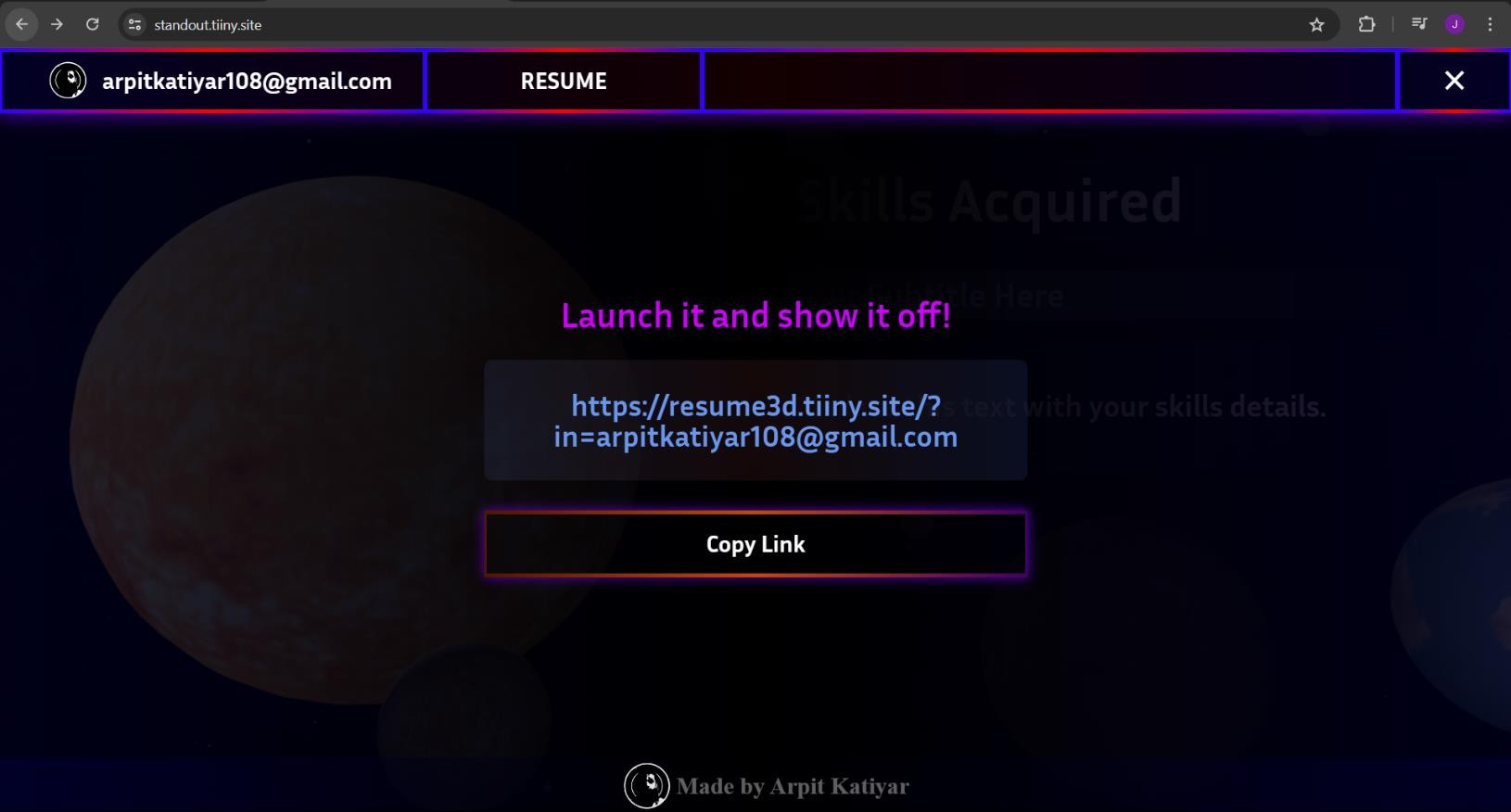
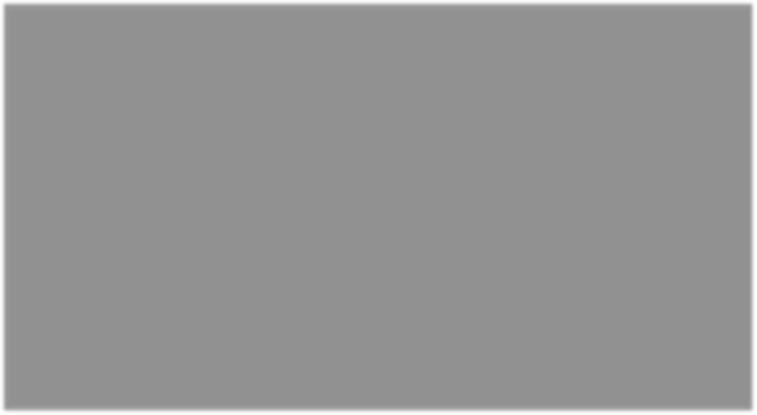
**THERE ARE 6 OTHER PAGES LIKE SO:**



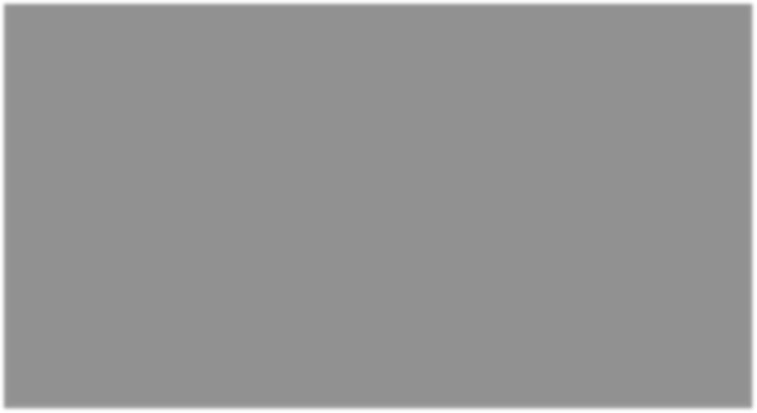
**LAUNCH PROFILE MENU:**



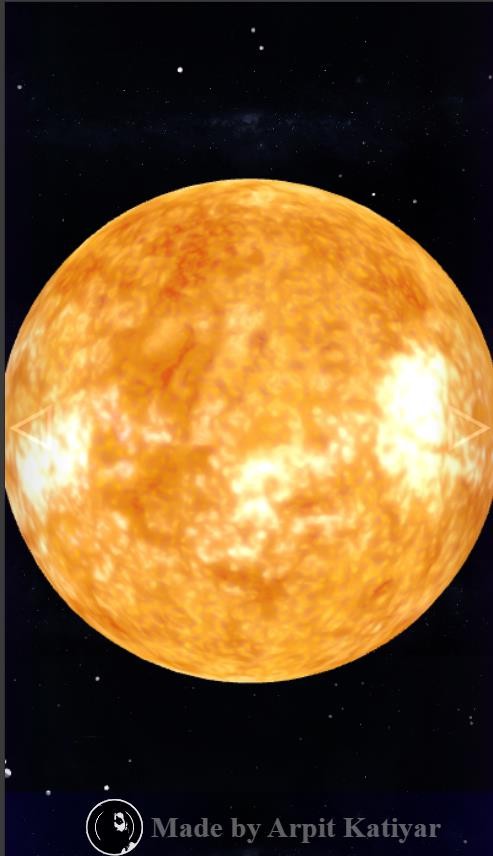
**SHARE LINK PAGE:**

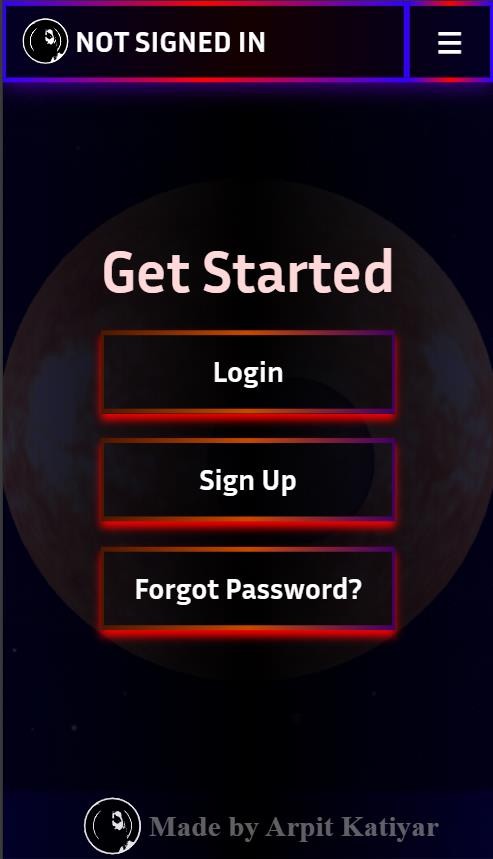


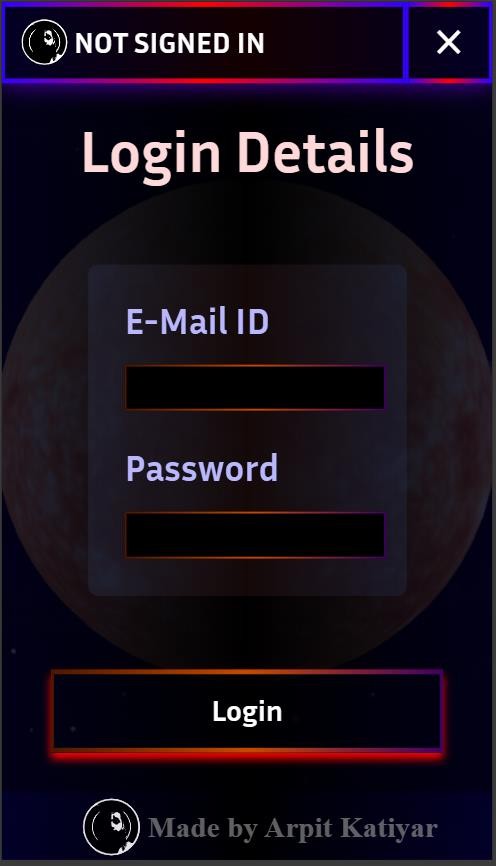
**OUTRO:**

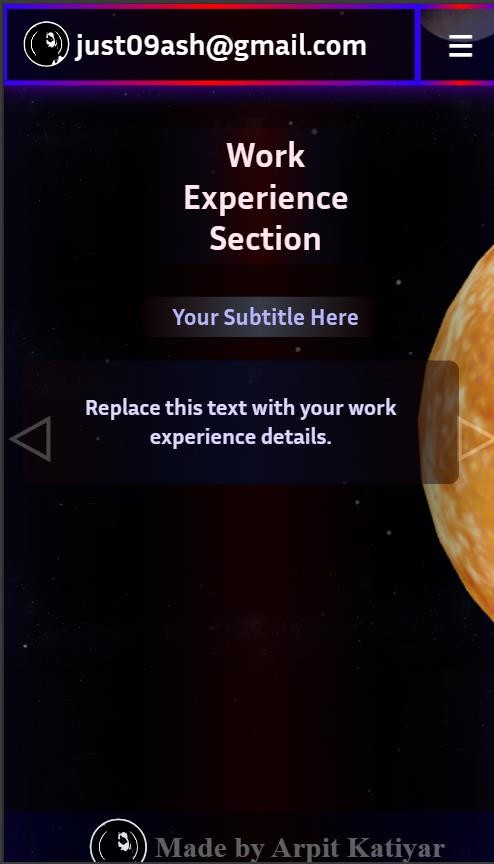


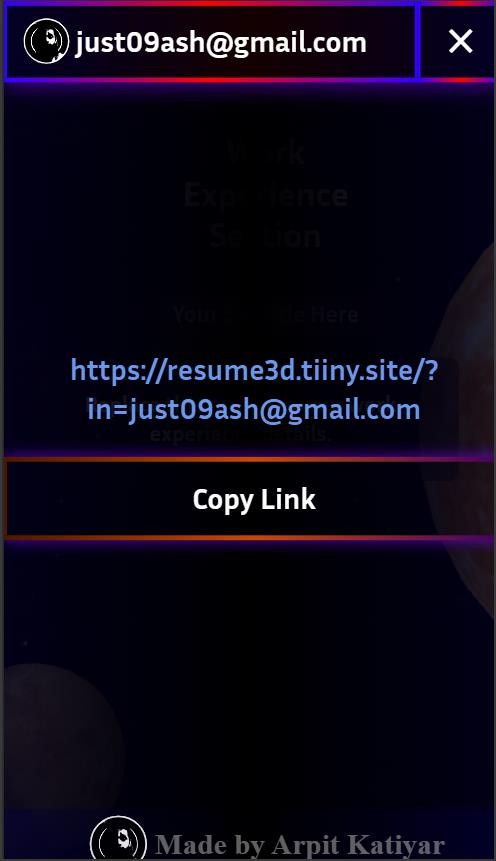
**MOBILE RESPONSIVE #1:**

**MOBILE RESPONSIVE #2:**

**MOBILE RESPONSIVE #3:**

**MOBILE RESPONSIVE #4:**

**MOBILE RESPONSIVE #5:**

**MOBILE RESPONSIVE #6:**

**MOBILE RESPONSIVE #7:**

# TESTING

##### Testing:

Testing process for the project, covering various aspects such as authentication, hosting, database operations, responsiveness, user interaction, and design, with explanations derived directly from Firebase documentation and other relevant sources:

1. Authentication via Firebase OAuth:
   * In accordance with Firebase documentation, extensive testing was conducted on user authentication mechanisms using Firebase Authentication.
   * This involved thorough examination of email and password authentication, ensuring secure sign-in functionality.
   * Testing also encompassed the implementation of OAuth authentication via Google, adhering to Firebase's recommended best practices for integrating OAuth providers.
   * Detailed checks were made to validate the handling of authentication tokens and user sessions, ensuring seamless user experiences while maintaining security standards.
2. Firestore Database Operations:
   * As per Firebase's Firestore documentation, comprehensive testing was carried out on Firestore database operations, including reading, writing, updating, and deleting data.
   * Each CRUD (Create, Read, Update, Delete) operation was meticulously tested to verify its functionality and efficiency.
   * Testing also involved the implementation and validation of security rules to control access to Firestore data, following Firebase's guidelines for secure data storage and access management.
   * Performance testing was conducted to assess the scalability and responsiveness of Firestore database operations under various load conditions, ensuring optimal performance even under high traffic scenarios.
3. Hosting and SEO Testing:
   * Following Firebase Hosting guidelines, the project was hosted on Firebase Hosting, with thorough testing performed to validate deployment and hosting configurations.
   * Firebase Hosting's SSL support and content delivery network (CDN) capabilities were evaluated to ensure secure and high-performance content delivery.
   * Testing included checks for Search Engine Optimization (SEO) aspects such as metadata configuration, canonical URLs, and structured data markup, aiming to improve visibility and ranking in search engine results pages (SERPs).
   * Page speed and performance metrics were analyzed using Firebase Hosting's integration with Google Analytics, with optimizations made based on insights derived from performance reports.
4. Responsiveness Testing:
   * In line with Firebase's recommendations for responsive web design, extensive testing was conducted to ensure optimal user experiences across various devices and screen sizes.
   * Testing involved the use of responsive design testing tools and emulators to simulate user interactions on different devices, including desktops, laptops, tablets, and smartphones.
   * Media queries and viewport settings were thoroughly tested to ensure consistent layout and styling across

different viewport sizes, in compliance with responsive design principles.

* + User interactions such as touch events and gestures

|  |  |  |  |
| --- | --- | --- | --- |
| were validated on | touch-enabled | devices, | with |
| adjustments made  accessibility. | to enhance | usability | and |

1. User Interaction Testing:
   * Firebase documentation on user interaction design principles served as a guide for testing user interaction features and functionalities.
   * Testing encompassed navigation patterns, input controls, form validation, feedback mechanisms, and other interactive elements, ensuring intuitive and user-friendly interactions.
   * Accessibility testing was conducted to assess compliance with web accessibility standards such as Web Content Accessibility Guidelines, with optimizations made to enhance usability for users with disabilities.
   * Usability testing sessions were conducted with representative users to gather feedback and identify usability issues, with iterative improvements made based on user feedback and usability testing results.
2. Design and UI Testing:
   * Drawing from Firebase's UI design principles and guidelines, thorough testing was performed on the project's design and user interface (UI) elements.
   * Visual design elements such as typography, color schemes, iconography, and layout compositions were evaluated for aesthetic appeal and visual consistency.
   * Usability testing sessions included tasks related to navigation, content discovery, and task completion, with observations recorded to identify design flaws and usability issues.
   * Cross-browser and cross-device testing was conducted to ensure compatibility and consistency of design rendering across different web browsers and devices.
   * Feedback from design reviews and usability testing sessions informed iterative design refinements, with revisions made to improve user experience and visual presentation.

Overall, the testing process involved meticulous attention to detail and adherence to best practices outlined in Firebase documentation and other relevant resources, with the goal of delivering a robust, secure, user-friendly, and visually appealing

web application that meets the project requirements and exceeds user expectations.

Black Box Testing for "StandOut 3D Resume Builder":

1. **Authentication Testing:**
   * Test user sign-in using email and password authentication.
   * Verify the authentication process via email for account verification.
   * Test the "forgot password" feature to ensure users can reset their passwords securely.
2. Database Operations Testing:
   * Test CRUD (Create, Read, Update, Delete) operations on Firestore database collections.
   * Verify the integrity and consistency of data stored in collections and documents.
   * Test data access permissions defined by Firestore security rules to ensure appropriate user access.
3. Hosting Testing:
   * Test website hosting on a chosen platform (e.g., Firebase Hosting) to ensure proper deployment.
   * Verify website accessibility across different devices and browsers.
   * Test website performance and responsiveness under various network conditions.
4. User Interaction Testing:
   * Test navigation features, such as on-screen buttons or arrow keys, for ease of use.
   * Verify user interactions for creating and editing 3D resumes, ensuring intuitive and seamless experiences.
   * Test sharing functionalities, such as generating unique shareable links for resumes.

White Box Testing for "StandOut 3D Resume Builder":

1. **Code Coverage Analysis:**
   * Perform code coverage analysis using tools like Istanbul to ensure adequate test coverage.
   * Verify that critical code paths, including authentication and database operations, are thoroughly tested.
2. Control Flow Testing:
   * Test different control flow scenarios in authentication processes, such as valid and invalid user inputs.
   * Verify control flow paths for error handling and exception scenarios.
3. Data Flow Testing:
   * Test data flow within the application, including data validation and sanitization.
   * Verify proper handling of user inputs to prevent security vulnerabilities like injection attacks.
4. Integration Testing:
   * Test integration between frontend (HTML, CSS, JavaScript) and backend (Firebase Authentication, Firestore Database) components.
   * Verify seamless communication between client-side and server-side code for real-time updates and data synchronization.

Testing Conclusion:

By combining black box and white box testing techniques, "StandOut 3D Resume Builder" can ensure both functional correctness and internal robustness. Black box testing focuses on validating user-facing features and interactions, while white box testing delves deeper into the internal logic and code structure. This comprehensive testing approach helps identify and address potential issues across all layers of the application, ultimately enhancing its quality, reliability, and user satisfaction.

# CONCLUSION

**Conclusion:**

**Reflections and Recommendations:**

In reflecting upon the journey of designing and testing "StandOut 3D Resume Builder," several key conclusions emerge, each shedding light on the application's strengths and areas for further enhancement. This conclusion encapsulates insights gleaned from the comprehensive black box testing process, offering recommendations for future iterations and user engagement strategies.

Automation Enhancements:

One of the foremost observations is the transformative impact of automation on the system's efficiency and reliability. By streamlining processes and minimizing manual interventions, the application demonstrates heightened responsiveness and scalability. Future enhancements should prioritize further automation to optimize user experiences and backend operations seamlessly.

Creative User Interface:

At the heart of "StandOut 3D Resume Builder" lies its creative and user-friendly 3D graphical interface. This innovative approach to resume building empowers users to craft visually captivating resumes effortlessly. Moving forward, continuous

refinement of the interface's design and functionality will be paramount to ensure an engaging and intuitive user experience.

Security Considerations:

While Firebase and Firestore offer robust security measures, it is essential to acknowledge the inherent limitations and potential vulnerabilities. While the application provides adequate security for typical usage scenarios, users are advised to exercise caution when inputting sensitive information. Adopting best practices, such as using unique passwords and refraining from sharing confidential data, mitigates potential risks and safeguards user privacy.

Authentication Mechanisms:

The integration of authentication mechanisms, including email and password authentication via Google services, ensures user data integrity and access control. However, users are reminded to exercise diligence in managing their account credentials and refrain from sharing sensitive information. Adopting a proactive approach to security awareness and user education will fortify the application's resilience against potential threats.

Future Directions:

Looking ahead, the evolution of "StandOut 3D Resume Builder" hinges on continuous innovation and user-centric design

principles. Future iterations should prioritize enhancements in automation, user interface refinements, and security fortifications. Additionally, fostering a community-driven feedback loop and prioritizing user engagement initiatives will foster a vibrant ecosystem of resume builders and job seekers.

In conclusion, the journey of designing and testing "StandOut 3D Resume Builder" has been marked by exploration, discovery, and growth. As we chart the course for the application's future, we remain committed to delivering a cutting-edge platform that empowers users to showcase their unique talents and aspirations effectively. With a steadfast focus on excellence and user satisfaction, "StandOut 3D Resume Builder" is poised to redefine the landscape of resume creation and presentation.

# FUTURE ENHANCEMENTS

**Future Enhancements and Announcements**

In anticipation of future iterations and enhancements for "StandOut 3D Resume Builder," a myriad of groundbreaking features and announcements are poised to enchance user experiences and propel the platform to new heights of progress:

1. Customizable Watermarks:
   * Embarking on a transformative journey towards greater user autonomy, the platform will introduce an innovative feature enabling users to tailor or eliminate watermarks bearing the creator's name, Arpit Katiyar. This groundbreaking initiative not only serves as a potential revenue stream but also empowers users to assert their branding preferences with unparalleled precision and finesse.
2. Expanded 3D Background Options:
   * Embracing diversity and creativity, the platform will unveil an expansive array of meticulously crafted 3D background environments available for purchase. Beyond the confines of space-themed backdrops, users will discover a treasure trove of captivating settings, each meticulously designed to captivate and inspire. By offering a curated selection of paid

background options, users will embark on a visual odyssey, elevating their resumes to unparalleled heights of sophistication and allure.

1. Pro Membership with Advanced Customization:
   * Pioneering a new era of customization and personalization, the platform will introduce an exclusive Pro membership tier boasting unparalleled customization capabilities. From fine-tuning the minutest details of the 3D scene to orchestrating a symphony of visual elements, users will wield unprecedented control over their resume presentations. This premium offering will empower users to transcend conventional boundaries, sculpting their resumes into masterpieces of ingenuity and distinction. Embracing font customization, color manipulation, and the integration of personal photos into the resume entry cube, the Pro membership promises a cornucopia of possibilities limited only by the bounds of imagination.
2. AI-Powered Grammar and Vocabulary Assistance:
   * Harnessing the transformative power of artificial intelligence, the platform will unveil an awe-inspiring AI chatbot feature engineered to elevate the quality of resume content to unprecedented heights. By

leveraging state-of-the-art natural language processing algorithms, users will embark on a journey of linguistic refinement, guided by an ever-vigilant digital mentor. From suggesting grammatically flawless prose to infusing narratives with a dash of lexical brilliance, the AI chatbot will serve as a beacon of inspiration, illuminating the path to resume perfection with unparalleled clarity and insight.

1. Continuous Updates and Bug Fixes:
   * Embracing a steadfast commitment to excellence and user satisfaction, the platform will inaugurate a relentless cycle of updates and bug fixes. Bolstered by a robust customer support system, users will be empowered to voice their concerns and provide invaluable feedback directly to the platform's custodians. Through a symbiotic partnership between users and developers, every glitch will be swiftly rectified, every suggestion meticulously considered. With each update, the platform will evolve, metamorphosing into a paragon of reliability, innovation, and user-centric design.

These monumental strides towards progress and innovation underscore the platform's unwavering dedication to empowering users, fostering creativity, and transcending the

boundaries of conventional resume building. As the platform continues to evolve and adapt to the ever-changing landscape of career development, one thing remains certain: the future is brimming with infinite possibilities, awaiting exploration and discovery.

# BIBILOGRAPHY

**BIBLIOGRAPHY:**

Bibliography

**Firebase Authentication and Firestore Data Management: Net Ninja**

YouTube Channel: [Net Ninja](https://www.youtube.com/channel/UCW5YeuERMmlnqo4oq8vwUpg)

Tutorials: Firebase integration, authentication mechanisms, Firestore data management

3D Visualization with Three.js:

**Fireship**

YouTube Channel: [Fireship](https://www.youtube.com/channel/UCsBjURrPoezykLs9EqgamOA)

Tutorials: Three.js fundamentals, immersive 3D web development

Wael Yasmina

YouTube Channel: [Wael Yasmina](https://www.youtube.com/%40WaelYasmina)

Tutorials: Advanced Three.js techniques, 3D visualization concepts

HTML, CSS, and JavaScript Mastery:

**W3Schools**

Website: [W3Schools](https://www.w3schools.com/)

Resources: Comprehensive tutorials on HTML, CSS, JavaScript

AI-driven Platforms:

**OpenAI ChatGPT**

Website: [OpenAI](https://openai.com/chatgpt)

Platform: AI-driven assistance for coding challenges and refinement

Google Gemini

Platform: AI-powered assistant for coding and problem-solving

Microsoft CoPilot

Platform: AI-driven code completion and suggestion tool

In Conclusion,

The journey of crafting the "StandOut 3D Resume Builder" was inspired by a multitude of creators and resources. It served as a testament to the transformative power of internet, innovation, and creative endeavor.

- ARPIT KATIYAR

