**Imagination-Station**

Imagination-Station is a storytelling platform that transforms kids’ dreams and imaginations into a digital, illustrative storybook. Built on a Django REST Framework back-end, a React + Vite front-end, and a PostgreSQL database, it provides an engaging, 3D-animated interface where children can create, visualize, and share their stories with family and friends.

**Table of Contents**

1. [Features](#features)
2. [Technology Stack](#technology-stack)
3. [Installation & Setup](#installation--setup)
4. [Project Structure](#project-structure)
5. [Usage](#usage)
6. [Roadmap](#roadmap)
7. [Contributing](#contributing)
8. [License](#license)

**Features**

* **Child-Friendly UI**  
  Large buttons, bright colors, and simple layouts designed with young users in mind.
* **Django RESTful API**  
  A robust backend for handling user data, story content, and 3D model references.
* **React + Vite Front-End**  
  Fast, modern UI/UX with dynamic page transitions using Framer Motion and 3D animations via @react-three/fiber and @react-three/drei.
* **Story Creation Tools**  
  A form-driven approach letting children write or dictate their stories, attach images, and choose from a growing 3D model library to illustrate scenes.
* **3D Scene & Animations**  
  Real-time 3D rendering with Three.js, showcasing playful animations like floating objects and orbit controls to engage children visually.
* **Database & Persistence**  
  Powered by PostgreSQL to store user info, story progress, and media, ensuring reliable data management.

**Technology Stack**

* **Front-end**: React + Vite, Tailwind CSS, Framer Motion, @react-three/fiber, @react-three/drei
* **Back-end**: Python, Django, Django REST Framework
* **Database**: PostgreSQL
* **3D Rendering**: Three.js (via @react-three/fiber)

**Installation & Setup**

**Prerequisites**

* Python 3.9+
* Node.js 16+
* PostgreSQL installed and running

1. **Clone the Repository**
2. git clone https://github.com/yourusername/imagination-station.git
3. cd imagination-station
4. **Back-End (Django + DRF Setup)**
   * Create and activate a virtual environment:
   * python3 -m venv venv
   * source venv/bin/activate
   * Install dependencies:
   * pip install -r requirements.txt
   * Configure your PostgreSQL credentials in settings.py (or using environment variables):
   * DATABASES = {
   * 'default': {
   * 'ENGINE': 'django.db.backends.postgresql',
   * 'NAME': 'imagination\_db',
   * 'USER': 'postgres\_user',
   * 'PASSWORD': 'postgres\_password',
   * 'HOST': 'localhost',
   * 'PORT': '5432',
   * }
   * }
   * Run migrations and start the server:
   * python manage.py migrate
   * python manage.py runserver
5. **Front-End (React + Vite Setup)**
   * From the project root, navigate to the frontend folder (assuming your React app is in frontend/):
   * cd frontend
   * Install front-end dependencies:
   * npm install
   * Start the development server:
   * npm run dev
   * The app is typically served at <http://localhost:5173>, but check your terminal for the exact port.

**Project Structure**

Below is a simplified overview:

imagination-station/

├── backend/

│ ├── imagination/ # Django project folder

│ ├── manage.py

│ ├── requirements.txt

│ └── ...

├── frontend/

│ ├── src/

│ │ ├── assets/

│ │ │ ├── Components/ # React components (Hero, Controls, etc.)

│ │ │ ├── models/ # 3D model components or .glb files

│ │ ├── App.jsx

│ │ └── main.jsx

│ ├── index.html

│ ├── package.json

│ └── ...

└── README.md

* **backend/imagination**: Contains your Django configurations, apps, and REST Framework serializers.
* **frontend/src**: Holds all front-end React code, components, hooks, 3D objects, etc.

**Usage**

1. **Register / Log In**  
   After launching the server, navigate to the front-end URL. Create a new user or log in with your existing account.
2. **Create a Story**
   * Click the **"Create Story"** button to start drafting a new story.
   * Input a title, story text, and choose your desired 3D elements. Kids can illustrate their ideas using the 3D model library.
3. **View 3D Scenes**
   * The 3D environment will display a playful background with floating or moving models.
   * You can rotate the camera (OrbitControls) or watch the animations with the Float effect.
4. **Edit / Update**
   * Stories can be edited at any time. Once satisfied, the final story can be viewed in a “book-like” layout or an interactive 3D environment.
5. **Explore Other Stories**
   * Browse or search for stories created by other kids (if public sharing is enabled) to get inspiration or just for fun reading.

**Roadmap**

* **Real-Time Collaboration**: Allow multiple users to co-edit the same story simultaneously.
* **Audio Storytelling**: Include a voice narration feature or text-to-speech for inclusive reading experiences.
* **Interactive Mini-Games**: Integrate small puzzles or story-driven mini-games.
* **Community Sharing & Moderation**: Introduce safe, moderated social features for kids to share feedback on each other’s stories.
* **Enhanced 3D Library**: Add more child-friendly models and interactive props.
* **Educational Tools**: Gamify reading comprehension, grammar checks, and vocabulary building.

**Contributing**

Contributions are welcome! Please follow these steps:

1. **Fork** the repository and clone it locally.
2. Create a **branch** for your feature:
3. git checkout -b feature/my-new-feature
4. **Commit** your changes with a descriptive message.
5. **Push** the branch to your fork:
6. git push origin feature/my-new-feature
7. Create a **Pull Request** from your feature branch into the main repository.

We appreciate bug reports, suggestions, and any improvements you can offer!