Project 2
StoryQuest - Interactive Storytelling Adventure

1 Project Introduction

StoryQuest is an interactive storytelling adventure game where players make choices that shape the storyline. Each decision leads to different branches and multiple endings, encouraging replayability.
Players can track their story path, save progress, and discover alternate outcomes. The game is built using React with Hooks, follows the Functional Programming Paradigm, and stores progress using Firestore.

2 Business Requirements

Goal:
StoryQuest is an interactive storytelling adventure game where players make choices that shape the storyline. Each decision leads to different branches and multiple endings, allowing users to explore various paths and outcomes. The game will save user progress, track their choices, and provide a visual representation of the story path, encouraging replayability.

Key Features:
- **Story Selection** – Players can choose from multiple story genres (e.g., fantasy, sci-fi, mystery).
- **Branching Choices** – Every decision affects the storyline, leading to different paths and endings.
- **Multiple Endings** – Users can replay the story to explore alternate outcomes.
- **Progress Tracking** – Saves user choices and allows them to continue from their last decision.
- **Story Path Visualization** – A simple UI showing the choices taken and how they branch.
- **Firestore Storage** – Stores user choices, progress, and unlocked endings.

Legend: 🟨 for nouns, 🟩 for verbs

3 Nouns-Verbs

Nouns:
- StoryQuest
- storytelling adventure game
- players
- choices
- storyline

- decision
- branches
- endings
- paths
- outcomes
- progress
- choices
- representation
- replayability
- story genres
- UI
- choices
- progress
- unlocked endings

Verbs:

- shape
- leads
- allow
- explore
- save
- track
- provide
- encourage
- choose
- affect
- replay
- alternate
- saves
- continue
- showing
- stores

4 Target Audience

- Interactive fiction enthusiasts - Who enjoy making choices that impact the storyline.
- Casual gamers - Looking for an engaging yet simple adventure experience.
- Story-driven players - Who want to explore different endings.
- People who enjoy replayable content and choice-driven narratives.

5 Rules[1]

- Players start by choosing a story from a variety of genres.

- Each decision changes the storyline, leading to different branches and endings.
- Once a choice is made, it cannot be undone - players must play again to explore alternative paths.
- Progress is automatically saved, allowing players to continue later.
- Players can track their choices and endings through a visual story path.

6 Challenge Questions

- How can we design a modular story system so that new stories can be easily added?
- What is the best way to dynamically visualize branching paths?
- How can we optimize Firestore to effectively track user choices and progress?
- What features can increase player immersion and decision impact?
- How can we introduce unlockables to increase replayability?

7 Modules

- **App (Main Component)**
  Manages navigation and story selection.
  Associates with StoryManager when a story is chosen.

- **StoryManager**
  Controls story progression based on user choices.
  Aggregates StoryScreen for displaying story content.
  Associates with ProgressManager to load/save progress.

- **StoryScreen**
  Displays the current story scene.
  Aggregates ChoiceButtons for user decisions.

- **ChoiceButtons**
  Renders buttons for user choices.
  Calls StoryManager to update the story.

- **ProgressManager**
  Stores and retrieves user progress from Firestore.
  Associates with Firestore Database.

- **Firestore Database**
  Stores user choices and progress persistently.

8 Use Cases

User Personas & User Stories[2]

- **New Player - First-Time Explorer**

- Age: 25
- Background: A new player who is experiencing interactive storytelling for the first time. They want an intuitive and engaging way to start their adventure.
- Dimensions:
  Story Selection Intent: Wants an easy way to browse and start a story.
  Guidance Needs: Prefers a clear introduction to how the game works.
- User Stories:
  (1) Selecting a Story
  As a new player, I want to browse and select a story from a list so that I can begin my adventure with a storyline of my choice.
  (2) Understanding the Game Mechanics
  As a new player, I want a tutorial or instructions when I start my first story so that I can understand how to play the game.
  (3) Experimenting with Choices
  As a new player, I want to be able to see the immediate consequences of my choices so that I can learn how my decisions affect the story.
  (4) Restarting Easily
  As a new player, I want an easy way to restart the story so that I can try different choices without confusion.

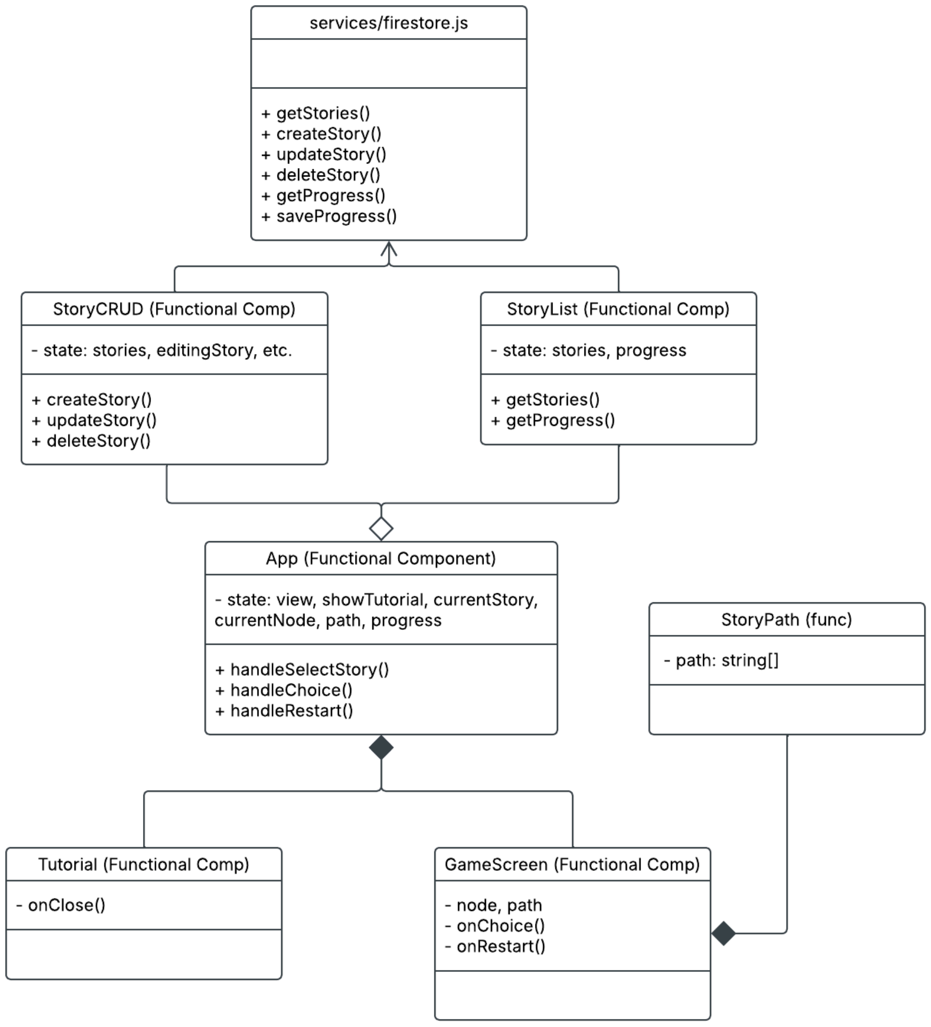
- **Returning Player - Story Enthusiast**
  - Age: 30
  - Background: A returning player who enjoys making decisions and seeing how they influence the story.
  - Dimensions:
    Decision-Making Experience: Wants choices that lead to meaningful changes in the story.
    Story Progression Interest: Likes to see the impact of their past decisions.
  - User Stories:
    (5) Making Choices to Advance the Story
    As a returning player, I want to make meaningful choices that impact the story so that I feel in control of the narrative.
    (6) Saving and Restoring Progress
    As a returning player, I want my progress to be saved automatically so that I can return later and continue from where I left off.
    (7) Discovering Hidden Storylines
    As a returning player, I want to explore hidden choices or secret paths so that I can uncover unexpected storylines.

- **Completionist - Ending Collector**
  - Age: 32
  - Background: A player who enjoys unlocking all possible endings and fully exploring the narrative.

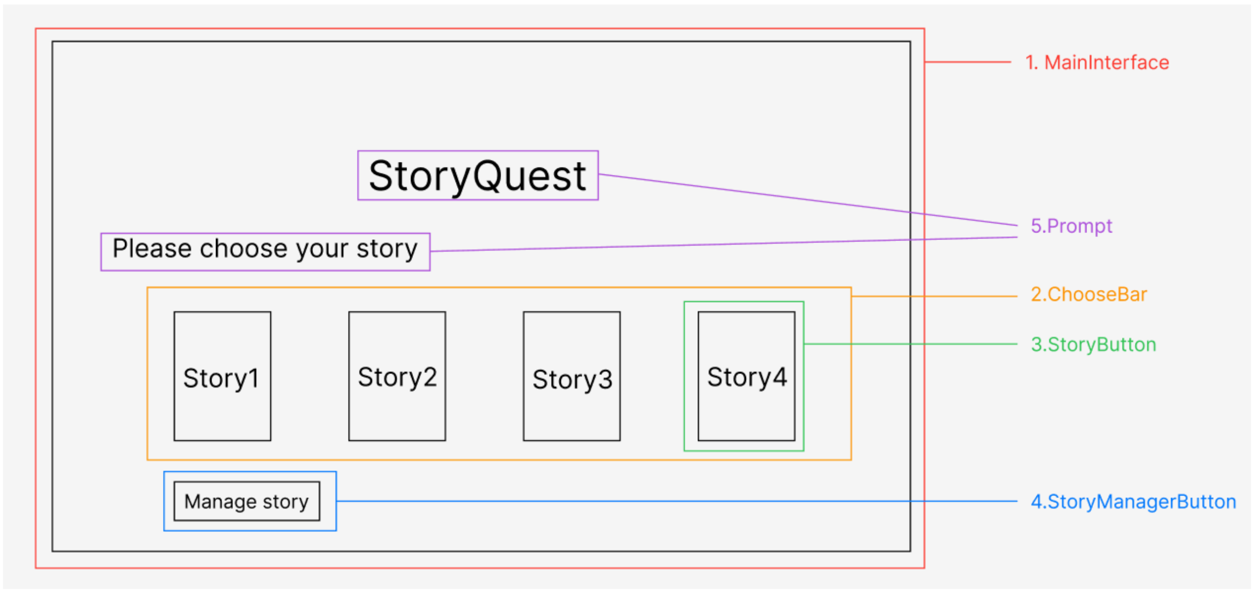
- Dimensions:
  Replayability Interest: Wants to replay the game and explore different story branches.
  Completionist Goals: Likes to track all unlocked endings and paths.
- User Stories:
  (8) Viewing Story Paths and Choices
  As a completionist, I want to see a visual representation of my choices so that I can understand how my decisions influenced the story.
  (9) Unlocking Multiple Endings
  As a completionist, I want to replay the story and make different choices so that I can unlock alternate endings and fully explore the narrative.
  (10) Tracking My Progress
  As a completionist, I want a progress tracker that shows how many endings I've unlocked so that I know what's left to discover.

- **Administrator - Story Curator**
  - Age: 30
  - Background: A user responsible for managing all story-related content within the application. They oversee the creation of new stories, updates to existing ones, and overall story quality.
  - Dimensions
    Story Management: Focuses on adding, editing, or removing stories to keep the content fresh and accurate.
  - User Stories
    (11) Creating a New Story
    As an administrator, I want to be able to create a new story with multiple branches and endings so that I can continuously offer fresh and diverse content for players.
    (12) Editing Existing Stories
    As an administrator, I want to modify story details (title, description, nodes, endings) so that I can fix errors, refine storylines, or add new branches without disrupting the overall game experience.
    (13) Deleting Outdated or Irrelevant Stories
    As an administrator, I want to remove old or irrelevant stories so that the story library remains relevant, up-to-date, and uncluttered.
    (14) Monitoring Story Engagement
    As an administrator, I want to see which stories are most popular or have the highest completion rates so that I can understand user preferences and optimize the storytelling experience.

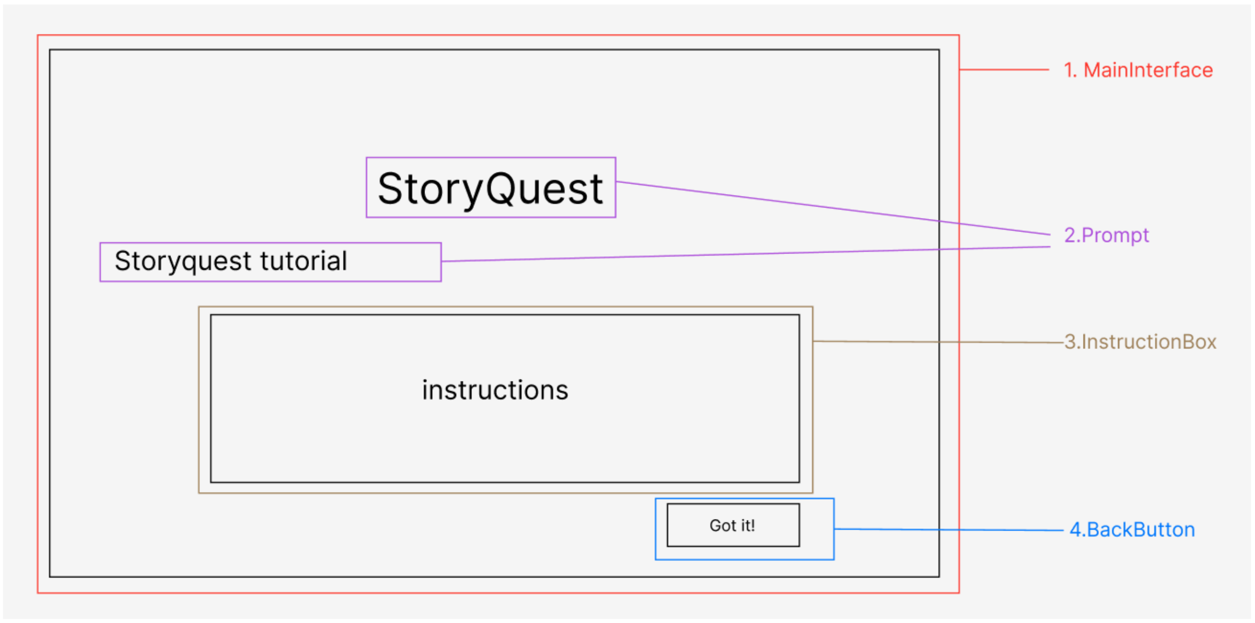
9 Modules Diagram



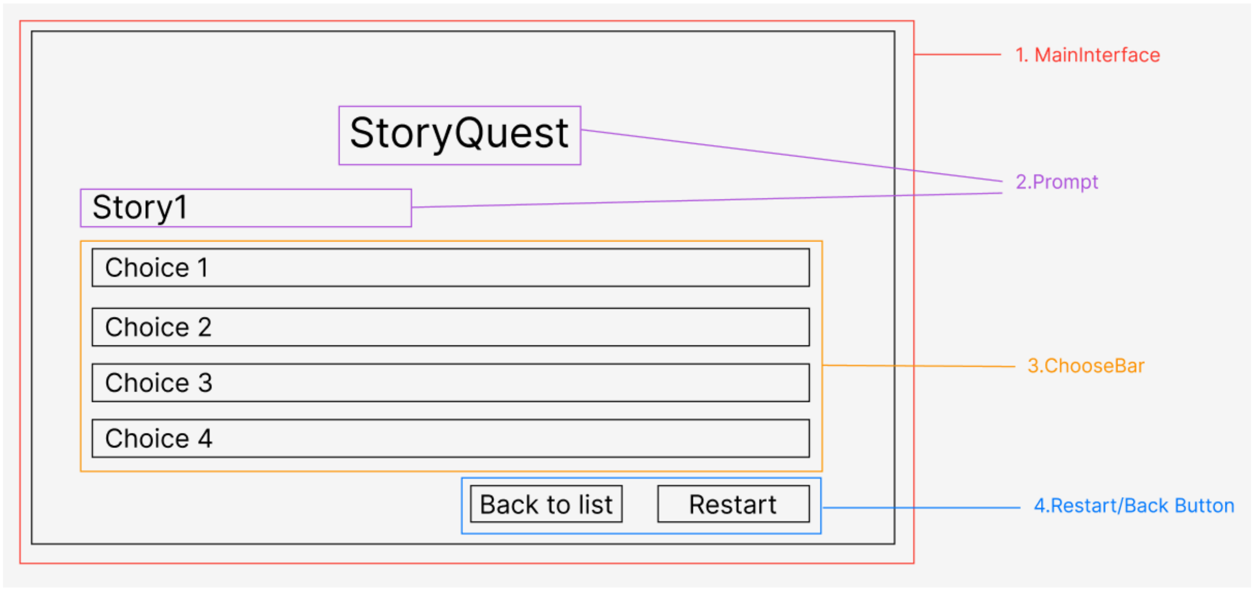
10 Interface Mockups



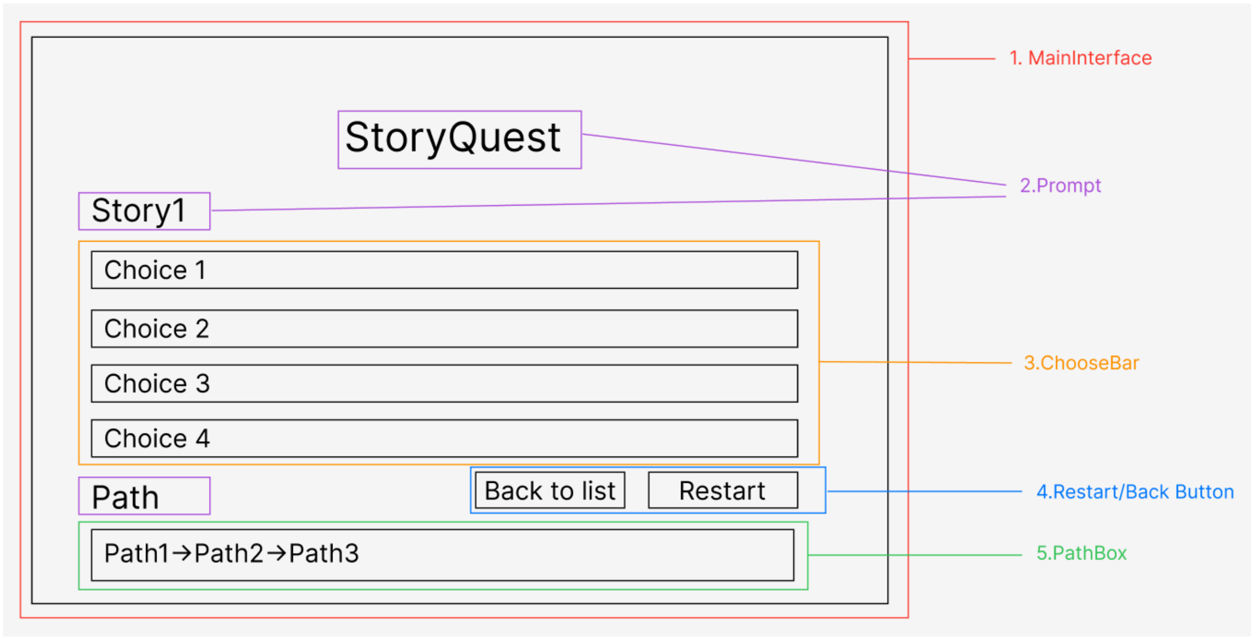
User story 1 - Start the game



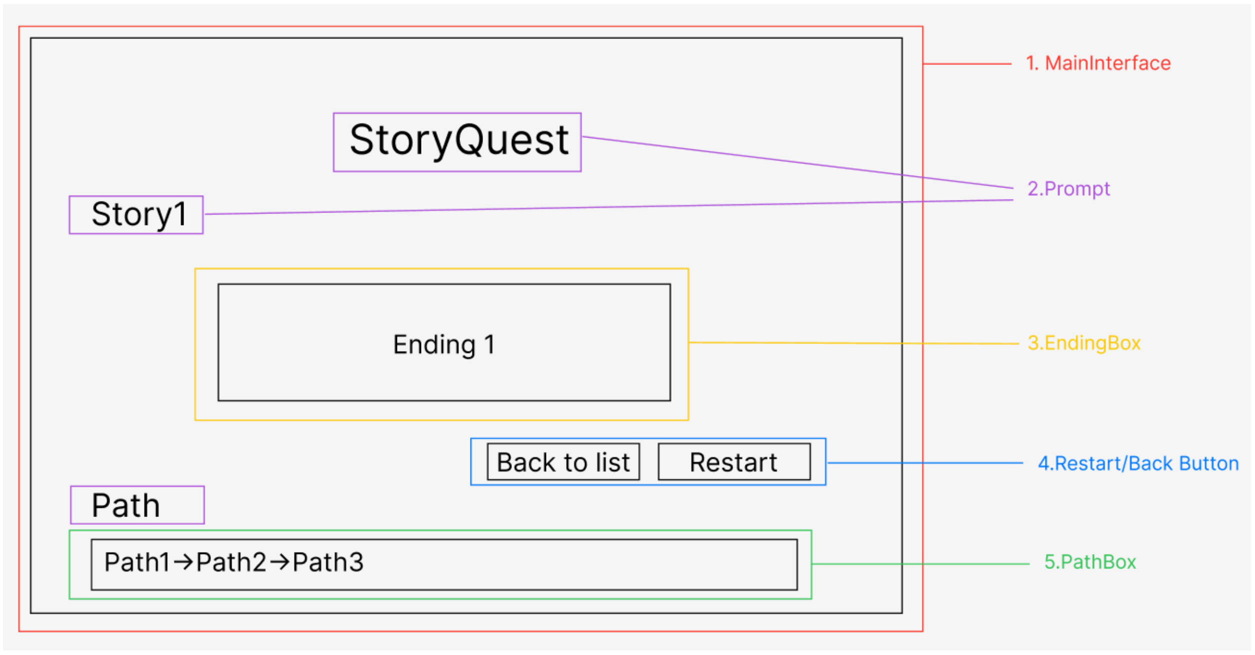
User story 2 - Game tutorial and instructions



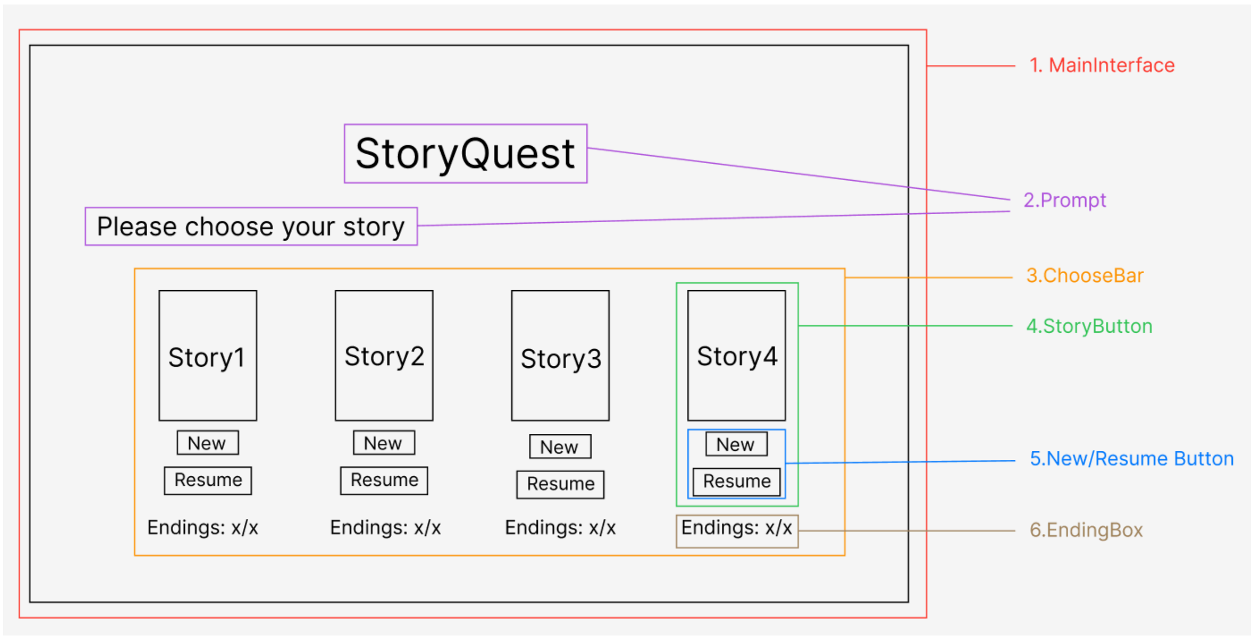
User story 3, 4 - Make the choice and restart



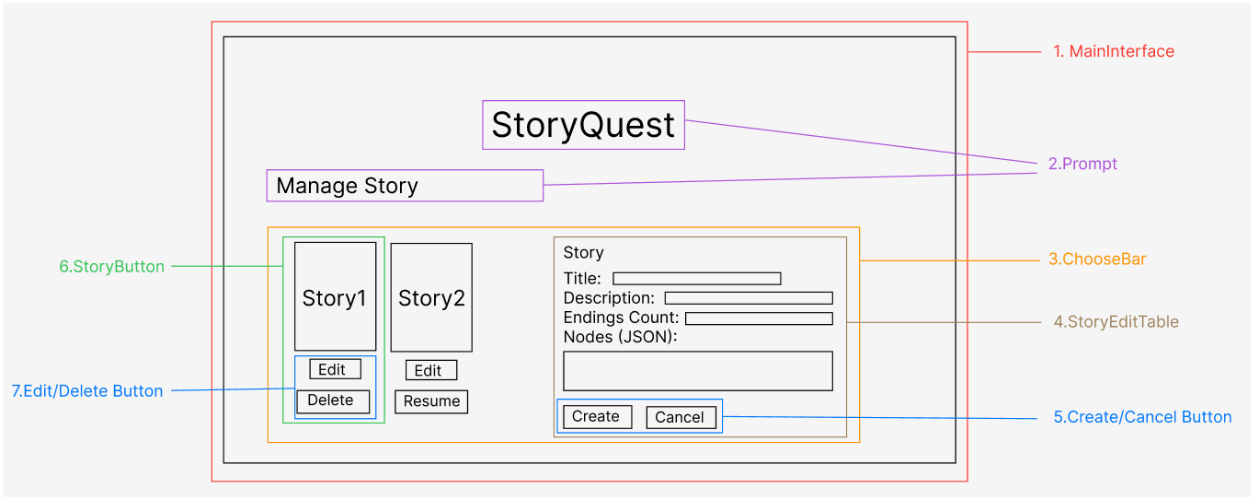
User story 5, 7 - Make the choice by past paths



User story 8, 9 - Game Ending



User story 6, 10 - Check the status of stories and new/resume game



User story 11-14 - Manage(CRUD) the stories

11 Links

Module Diagram:
https://lucid.app/lucidchart/0b3de2d8-1ebf-47ab-ac12-2ff5906f0b67/edit?viewport_loc=-432%2C-76%2C2217%2C1092%2C0_0&invitationId=inv_74cef52b-fd53-432a-b1df-80f6ee463989

Interface Mockups:
https://www.figma.com/design/C2wUCrvW3aC10LOcClWjoK/Project2?node-id=0-1&t=6GffqFLwBUzs66Sd-1

12 Acknowledgment

I used ChatGPT4 to help me build this project and clarify some new concepts and methods as well as generate the examples.

Prompt1:

There is my business requirement, can you help me to generate the rules?

Response:

There are some rules

- Players start by choosing a story from a variety of genres.
- Each decision changes the storyline, leading to different branches and endings.
- Once a choice is made, it cannot be undone - players must play again to explore alternative paths.

  …

Check:
I checked the answer and select some rules. I also add some rules of this part.

Prompt2:
Can you help me to generate some user stories for this persona?
New Player - First-Time Explorer
Age: 25
Background: A new player who is experiencing interactive storytelling for the first time. They want an intuitive and engaging way to start their adventure.
Dimensions:
Story Selection Intent: Wants an easy way to browse and start a story.
Guidance Needs: Prefers a clear introduction to how the game works.

Response:
There are some user stories:
(1) Selecting a Story
As a new player, I want to browse and select a story from a list so that I can begin my adventure with a storyline of my choice.
(2) Understanding the Game Mechanics
As a new player, I want a tutorial or instructions when I start my first story so that I can understand how to play the game.
(3) Experimenting with Choices
As a new player, I want to be able to see the immediate consequences of my choices so that I can learn how my decisions affect the story.

Check:
I checked these user stories and do some modifications and add more stories of this personas.

Prompt3:
How can I build a react app?

Response:
Building a React app involves several steps, from setting up your development environment to writing and running your React components. Here’s a step-by-step guide:

1. Install Node.js and npm

React requires Node.js and npm (Node Package Manager) to manage dependencies.

Download and install Node.js from nodejs.org
Verify installation by running:
node -v
npm -v

2. Create a New React App
The easiest way to create a new React app is using Vite (faster) or Create React App (CRA).

Option 1: Using Vite (Recommended)
Vite is much faster and more efficient.
…

Check:
I learned how to build a react app and follow the instructions to build a react frame for this project.

Prompt4:
What is the Firestore and how to use it?

Response:
What is Firestore?
Firestore (also known as Cloud Firestore) is a NoSQL, cloud-hosted database from Firebase (Google Cloud Platform) designed for scalable and real-time applications.

Key Features of Firestore
NoSQL Document Database - Data is stored in documents and collections instead of tables and rows.
Real-Time Synchronization - Automatically updates data across connected devices.
Scalability - Handles large-scale applications with ease.
Offline Support - Data can be cached and accessed even when offline.
…

Check:

I learned the concept of Firestore and how to use this tool. It will help me to build this project and store the data easily.

Prompt5:(Firestore connection part)
How can I connect my vite+react project with Firestore and read/write the data?

Response:
Below is a concise guide on how to connect a Vite + React project with Firestore (Firebase's NoSQL database) and perform basic read/write operations. We'll use the Firebase v9 (modular) SDK.

1. Install Firebase
In your React (Vite) project directory, install the Firebase SDK:

```
npm install firebase
```

2. Create a Firebase Project & Grab Config
Go to the Firebase Console, create a new project (or use an existing one).

In Project Settings → General tab, under "Your apps," you'll see your Firebase configuration object, which looks like:

```
const firebaseConfig = {
  apiKey: "YOUR_API_KEY",
  authDomain: "YOUR_AUTH_DOMAIN",
  projectId: "YOUR_PROJECT_ID",
  storageBucket: "YOUR_STORAGE_BUCKET",
  messagingSenderId: "YOUR_MESSAGING_SENDER_ID",
  appId: "YOUR_APP_ID"
};
```

…

Check:
I followed these instructions to connect them and it works.

Prompt6:(Layout part)
How can I center my components of the whole web page?

Response:
To center your entire web page and all its components in a Vite + React project, follow these steps:

Method 1: Use Flexbox (Recommended)
Modify your global CSS file (e.g., index.css or App.css) to make the entire page center-aligned:

```
/* index.css or App.css */
html, body {
  margin: 0;
  padding: 0;
  height: 100%;
  display: flex;
  justify-content: center; /* Centers horizontally */
  align-items: center; /* Centers vertically */
  min-height: 100vh; /* Ensures full height */
  background-color: #f4f4f4; /* Optional */
}
```

Then, wrap your React app inside a container in App.jsx:

```
function App() {
  return (
    <div className="app-container">
      <h1>StoryQuest</h1>
      <p>Choose your adventure</p>
      {/* Other components */}
    </div>
  )
}
```

Check:
I followed these steps to center my components and it works.

Prompt7:(Data upload part)

Response:

Write a Batch Upload Script
Create a script (e.g., uploadStories.js) inside a services folder to push multiple stories at once.

```
// services/uploadStories.js
import { db } from "../firebaseConfig"; // Import Firestore instance
import { collection, writeBatch, doc } from "firebase/firestore";

// Sample batch of stories
const stories = [
…
];

async function uploadStories() {
  const batch = writeBatch(db); // Create a batch operation
  const storiesCollection = collection(db, "stories"); // Reference Firestore collection

  stories.forEach(story => {
    const storyRef = doc(storiesCollection, story.id); // Create document reference
    batch.set(storyRef, story); // Add story to batch
  });

  try {
    await batch.commit(); // Commit all writes at once
    console.log("Stories uploaded successfully!");
  } catch (error) {
    console.error("Error uploading stories:", error);
  }
}

// Run the script
uploadStories();
```

Check:
I followed these steps and modified some code to fit my project that can work well. I also added some new stories so that it can be shown on the page.