Tab 1

**GDG INNOHACKS**

**Hackathon Theme**

Finding the exact solution by using the resources that are provided.

**Domains**

Finalized domains are...

* AI[DS&ML] Hard Data --> Classif.
* Web Dev Job Portal
* Power BI Destroyed Data Set
* UI/UX Food Restaurant

**Timeline of Hack**

There will be two sessions in this hackathon and each with 2 hours.

* Intro Session 10:00 AM - 10:30 AM
* Session 01 10:30 AM - 12:30 PM
* Break 12:30 PM - 01:30 PM
* Session 02 01:30 PM - 03:30 PM
* Declaration 03:00 PM - 04:30 PM

**Team Size**

The Team Size will be min. 2 to max. 5.

**Jury Panel**

The Jury panel will be a team of 4 members, consisting of

4 Winners - [ *2 Faculty + 2 Students X 4 ].*

* { Principal, HoD’s }
* AI[DS & ML] - { + Dileep Sathya, Greeshma }
* Web Dev - { + Sisindri Singamsetti, Dishika }
* Power BI - { + A Anirudh, Eswar }
* UI/UX - { + Gnana Sekhar, Shashank }

**Incharges**

M Eswar Kumar [ Head ]

Sisindri Singamsetti [ Co-Head ]

**Resources that are provided:**

Images, Data Sets, AI Tools, Online Sites, Animation Sites etc..,

### 

### 

### **Web Dev**

**Incharges:**

1. **Sisindri Singamsetti**
2. **Dishika Vaishkiyar**

### **Question: The Invisible Job Board**

🔍 **web dev:**In an era where job seekers are inundated with an overwhelming number of listings—many of which do not align with their expertise—a groundbreaking hiring platform is introducing an innovative mechanism that **restricts access to job opportunities** unless candidates can substantiate their relevance. The philosophy behind this paradigm shift is that only individuals who demonstrably possess the requisite skill set should be granted visibility into potential career prospects.

Your objective is to architect a **highly interactive job portal** that strictly adheres to the following principles:

* **Complete Concealment:** Job listings must remain imperceptible by default, ensuring that users are unable to browse any positions without demonstrating competency.
* **Skill-Driven Discovery:** The portal should prompt users to manually input their proficiencies. As the user enters skills, job postings that correlate with at least one of the provided competencies should **gradually materialize** on the interface.
* **Relevance-Based Prioritization:** Any position exhibiting an alignment of **80% or greater** with the user's skill set should be prominently marked as a **"Top Match."**
* **Adaptive Ordering Mechanism:** Job postings must be arranged in a hierarchical fashion, with the most relevant opportunities surfacing at the forefront.

To enhance the sophistication and user experience of the system, consider integrating:

* **Intelligent Predictive Assistance:** A responsive input field that anticipates and suggests skills as users type, leveraging a precompiled repository of industry-relevant expertise.
* **Seamless Visual Transitions:** Implement elegant visual effects that orchestrate a **smooth unveiling** of job postings rather than an abrupt appearance.
* **Optimized Accessibility:** The platform should be meticulously engineered to ensure that its interface maintains fluid adaptability across a multitude of screen dimensions, particularly for mobile users.

❗ **Final Challenge:** At no point should users have the ability to view any job listings unless they actively specify at least one qualifying skill. The mechanism facilitating this revelation must operate autonomously, eliminating the need for explicit "search" interactions such as buttons or manual triggers.

🔑 **Considerations to Navigate the Challenge:**

* Evaluate how **dynamic filtering methodologies** can be leveraged to enhance data visibility while maintaining logical constraints.
* Contemplate an approach where **structured mapping of proficiencies to job descriptions** enables an intuitive yet computationally efficient matching process.
* Design an architecture where **real-time event-driven interactions** seamlessly govern the job list’s adaptability in response to user input.

This challenge necessitates a **nuanced understanding of frontend development principles**, compelling participants to think critically about **state management, input handling, and real-time UI updates**.

### 

### **🛠 Essential Resources & Tools for Development**

#### **1️. Frontend Frameworks & Libraries (For UI & State Management)**

Since this is a **frontend-only project**, you need tools to manage dynamic content, UI interactions, and responsiveness.

* **Primary Choice:**
  + [React.js](https://react.dev/) – Efficient for building dynamic UI with state management.
* **Alternatives:**
  + [Vue.js](https://vuejs.org/) – Progressive framework with built-in reactivity.
  + [Svelte](https://svelte.dev/) – Simple syntax and compiled at build time for better performance.

#### **2️. UI Styling & Component Libraries**

For building an interactive, modern, and responsive UI.

* **Primary Choice:**
  + [Tailwind CSS](https://tailwindcss.com/) – Utility-first CSS framework for fast UI development.
* **Alternatives:**
  + [Bootstrap](https://getbootstrap.com/) – Pre-styled components and grid system.
  + [Material UI](https://mui.com/) – Google's Material Design components for React.

### **🌐 APIs for Content Rendering**

To dynamically fetch **job listings, skills, and job descriptions**, you can use APIs to populate the page.

#### **3️. Job Listings APIs**

Fetching real-world job data dynamically.

* **Primary Choice:**
  + [Indeed API](https://www.indeed.com/developer) – Provides real-time job postings.
* **Alternatives:**
  + [LinkedIn Jobs API](https://developer.linkedin.com/docs/rest-api) – Retrieves job data from LinkedIn.
  + [Google Jobs API](https://cloud.google.com/talent-solution/job-search) – AI-powered job search API.

#### **4️. Skills Dataset & AI-Powered Search**

Since the challenge involves **predicting and matching skills dynamically**, these APIs will help:

* **Primary Choice:**
  + [O\*NET Skills API](https://services.onetcenter.org/) – Database of jobs, descriptions, and skills.
* **Alternatives:**
  + [Algolia API](https://www.algolia.com/) – AI-powered search API for skill suggestions.
  + [Skill List Dataset](https://github.com/tombatossals/skillicons) – Open-source tech skills database.

**5️. AI-Based Autocomplete & Predictive Assistance**

To suggest relevant skills in real time.

* **Primary Choice:**
  + [Claude AI](https://claude.ai/) – Can generate predictive suggestions for relevant skills.
* **Alternatives:**
  + [OpenAI GPT API](https://openai.com/) – Can be used for skill recommendation.
  + [Google Cloud Natural Language API](https://cloud.google.com/natural-language) – AI-powered text analysis.

### **🔄 Dynamic Content Handling & Rendering**

To process, filter, and display job postings dynamically.

#### **6️. JavaScript Libraries for Filtering & Sorting**

Since job listings should **dynamically appear based on input**, these will help:

* **Primary Choice:**
  + [Lodash.js](https://lodash.com/) – Provides filtering, sorting, and data manipulation utilities.
* **Alternatives:**
  + [Ramda.js](https://ramdajs.com/) – Functional programming for real-time filtering.
  + [D3.js](https://d3js.org/) – Can be used for visualizing job data.

#### **7️. Smooth UI Transitions & Animations**

Since job postings should gradually **appear smoothly**.

* **Primary Choice:**
  + [Framer Motion](https://www.framer.com/motion/) – Best for smooth animations in React.
* **Alternatives:**
  + [GSAP](https://greensock.com/gsap/) – Advanced animations for a polished UI.
  + [Animate.css](https://animate.style/) – Simple CSS-based animations.

### **🚀 AI Tools for Development & Debugging**

For faster coding, debugging, and content generation.

#### **8️. AI Code Assistants**

* **Primary Choice:**
  + [Claude AI](https://claude.ai/) – Helps generate optimized code, debug issues, and refine logic.
* **Alternatives:**
  + [ChatGPT](https://openai.com/chatgpt) – For generating code snippets and fixing bugs.
  + [GitHub Copilot](https://github.com/features/copilot) – AI-driven coding assistant.
  + [Codeium](https://www.codeium.com/) – Free alternative to Copilot.

#### **9️. Hosting & Deployment**

To make the project live.

* **Primary Choice:**
  + [Vercel](https://vercel.com/) – Best for **React-based** projects.
* **Alternatives:**
  + [Netlify](https://www.netlify.com/) – Best for static frontend hosting.
  + [GitHub Pages](https://pages.github.com/) – Free hosting for static sites.

### **💡 Alternative Approaches**

If you want to make the project even more interactive: ✅ Use **Web Speech API** for **voice-based skill input**.  
 ✅ Implement **Local Storage** to remember **user preferences**.  
 ✅ Experiment with **Lottie animations** for **revealing job postings smoothly**.

These **resources, APIs, and AI tools** will help you build, **fetch job data, filter results, animate UI, and deploy the project** successfully. Let me know if you need help implementing any part of it! 🚀

# **Action items**

* **Complete the designing**
* **Complete within the deadline**

**Mail the notebook to** [**Sisindri Singamsetti**](mailto:sisindrisingamsetti@gmail.com)

## 

## **UI/UX**

**Incharges:**

1. **K Gnana Sekhar**
2. **Shashank**

**🍽️ The Vanishing Flavor – A UI/UX Design Challenge**

Once renowned for its **signature dish, "The Eternal Feast"**, a famous restaurant’s website has mysteriously lost all traces of its menu overnight. Guests now arrive confused, unsure of what to order.

The restaurant owner claims that **only those who truly appreciate flavors will be able to restore the lost menu**. Your mission?  
✨ **Design a visually engaging and interactive restaurant landing page in Figma** that gradually **reveals the missing menu** through user interaction.

### **🛠️ UI/UX Design Requirements**

Your webpage should include **four visually engaging sections** that make the user feel like they are uncovering a **culinary mystery**.

**1️. The Fading Welcome Section**

* The page opens with a **blurred-out restaurant name** and tagline.
* As the user hovers over the title or scrolls, **letters gradually become clear**, revealing the restaurant’s identity.
* A **subtle animation effect** makes the background slightly dim until interaction begins.

**2️. The ‘Lost Menu’ Section**

* The menu appears **in a hidden or fragmented state**.
* Users must interact (e.g., hover, scroll, or click) to **uncover dishes one by one**.
* Each dish should have a **small animated effect**, like a steam rising animation or a glowing outline.

**3️. The Signature Experience Section**

* Introduce a **signature dish with an immersive visual** (e.g., a zoom-in effect on a plated dish).
* Users can **switch between "Ingredients" and "Story"** tabs to explore more details.
* A **subtle parallax effect** makes elements move slightly with user interaction.

**4️. The Final Invitation (Call to Action)**

* Users see a **"Dine with Us"** button that pulses slightly when hovered over.
* A small animation effect, like a **ripple effect or glowing edge**, should make the CTA feel **inviting**.
* **Bonus:** The button could have a **microinteraction** where it briefly reveals a secret dish name before clicking.

❗ **Final Challenge:** Users should never experience sudden jumps or static layouts. Every interaction must feel like **a carefully orchestrated dance of elements, revealing information progressively** without overwhelming the user.

## **🛠 Essential Resources for UI/UX Design**

To successfully build this **visually immersive restaurant website** in **Figma**, you need access to the right tools and assets.

### **1️. UI/UX Design Tools & Prototyping**

* **Primary Choice:**
  + [Figma](https://www.figma.com/) – Best for designing and prototyping animations.
* **Alternatives:**
  + [Adobe XD](https://www.adobe.com/products/xd.html) – Good for **transitions & interactions**.
  + [Framer](https://www.framer.com/) – Great for **advanced UI motion effects**.

### **2️. Animation & Motion Effects in Figma**

Since the challenge demands **smooth, elegant animations**, these plugins will help:

* **Primary Choice:**
  + [Figmotion](https://www.figma.com/community/plugin/733025261168520714/Figmotion) – Adds **motion effects** directly in Figma.
* **Alternatives:**
  + [Anima](https://www.animaapp.com/) – Converts Figma designs into **live animations**.
  + [Smart Animate (Built-in)](https://help.figma.com/hc/en-us/articles/360040028613-Create-animations-with-Smart-Animate) – For smooth **page transitions** and hover effects.

### **3️. Interactive Hover & Scroll Effects**

To make elements **dynamically appear** instead of staying static.

* **Primary Choice:**
  + [Scroll Animations](https://www.figma.com/community/plugin/1040933320409380591) – Adds **interactive reveal effects**.
* **Alternatives:**
  + [LottieFiles for Figma](https://www.figma.com/community/plugin/739058006757759323/LottieFiles) – For **realistic motion graphics**.
  + [Jitter](https://jitter.video/) – Creates **custom micro-interactions** for UI elements.

### **4️. High-Quality Restaurant UI Kits & Icons**

For **faster** and **more elegant designs**.

* **Primary Choice:**
  + [Foodie UI Kit](https://www.figma.com/community/file/1035155559530723152) – Ready-made food UI components.
* **Alternatives:**
  + [Flaticon Food Icons](https://www.flaticon.com/) – Free **restaurant-themed icons**.
  + [Icons8](https://icons8.com/) – High-quality **SVG icons & illustrations**.

### **5️. Fonts & Color Palettes**

A **luxury restaurant website** needs **elegant fonts and warm, inviting colors**.

* **Primary Choice:**
  + [Google Fonts (Playfair Display)](https://fonts.google.com/specimen/Playfair+Display) – Perfect for **sophisticated branding**.
* **Alternatives:**
  + [Adobe Fonts](https://fonts.adobe.com/) – Exclusive **high-end typography**.
  + [Coolors](https://coolors.co/) – Helps generate **perfect restaurant color schemes**.

### **6️. AI-Powered UI Assistance**

For **enhanced creativity and automatic UI generation**.

* **Primary Choice:**
  + [Claude AI](https://claude.ai/) – Helps refine **visual storytelling & layout choices**.
* **Alternatives:**
  + [ChatGPT for UI](https://openai.com/chatgpt) – Can suggest **Figma structure & best practices**.
  + [Uizard](https://uizard.io/) – Converts **sketches into UI designs automatically**.

### **⚡ Challenge Constraints**

✅ **All interactions should feel smooth and natural**—avoid jarring transitions.  
✅ **Use animations to guide the user, not distract them**.  
✅ **The design should be simple yet visually immersive**—focus on **storytelling through visuals**.  
✅ **Ensure responsiveness**—the design should adapt well to mobile screens.

# **Action items**

* **Complete the designing**
* **Complete within the deadline**

**Mail the notebook to** [**gnanasekharofficial@gmail.com**](mailto:gnanasekharofficial@gmail.com)

**Data Science**

**Incharges:**

1. **A Anirudh**
2. **Greeshma**

**Attachments**

[train](https://docs.google.com/spreadsheets/u/0/d/1fU0ccNC6rBvAlcnBgm_rj3IzECgBW_rrDk3pdQEzVMQ/edit) Dataset in Attachment

# **Question**

* Imagine you're a newly hired Data Science Manager at a prestigious organization. You're tasked with developing an employee promotion prediction model. Your boss demands exceptional performance and emphasizes that subpar results will not be tolerated.

Here are the cryptic clues to guide your quest:

| **Topic** | **File** |
| --- | --- |
| * Isolate your development environment to prevent contamination. | venv |
| * Consult the sacred texts: Features briefing, Pandas, NumPy, Scikit-learn, TensorFlow, PyTorch, and Keras cheat sheets. | [**pytorch-cheatsheet-en.pdf**](https://drive.google.com/open?id=1h5ZeNSkyuwkMRuPGE-TiN6YlR_dLTpvI) [**Pandas\_Cheat\_Sheet.pdf**](https://drive.google.com/open?id=15c8gPo-8gijc8omuXrLF621BXsOMMGWi) [**TensorFlow2Cheatsheet.pdf**](https://drive.google.com/open?id=1XQAHMB8YJ6U6sav6KUHhY_Tln0o7-owo) [**Keras\_Cheat\_Sheet\_gssmi8.pdf**](https://drive.google.com/open?id=1pyZweRBCX3swH4qcqZpyeV0oC_SJPT5r) [**Numpy\_Python\_Cheat\_Sheet.pdf**](https://drive.google.com/open?id=1B0ZU4l69InR_CafK8sRnFkF0EtU0ZDVR) |
| * Seek divine inspiration from Google and ChatGPT. |  |

# **Summary**

* At last the model should predict the output
* Any algorithm can be used
* Mode of approach should be mentioned in very first of the notebook 📒

# **Action items**

* Complete the modelling
* Complete within the deadline
* Mail the notebook to

# **Details**

### Raw notes

* Capture detailed flow of the modeling
* Predict the output
* Markdown the entire process step by step
* The accuracy score and confusion matrix score will be evaluated

**Power Bi**

**Incharges:**

1. **M Eswar Kumar**
2. **A Anirudh**

**Attachments**

* [citation.txt](https://drive.google.com/open?id=1r8tCX7VwkF9Rvf9AiP7unUKLXcD2oRAs)
* [IHME-GBD\_2021\_DATA-eef9b0d7-1.csv](https://drive.google.com/open?id=12Y-KpWWfzHm5_x9D4AnwtNIPkILVgwfo)

# 

# **Question**

* Analyze the Impact of Viruses on Mortality Rates Globally.

#### **Problem Statement:**

* Participants are tasked with creating a Power BI report to visualize and analyze the causes of death due to viruses. They should identify key insights about the mortality rates, trends, and regional impacts of specific viruses. The focus should be on presenting actionable insights using compelling visualizations.

| **Challenge Question:** | **Summary** |
| --- | --- |
| Top Causes of Death | * Identify and visualize the leading viruses causing death. |
| Trend Analysis: | * Show the trend of deaths caused by viruses over the years. Is there an increase or decrease? |
| Correlation Insights | * Explore potential correlations between mortality rates and other factors (if available in the dataset). |

# **Summary or key decisions**

* Summarize the most important outcomes
* Capture the trends/patterns

# **Action items**

* Complete the dashboard
* Complete within the deadline
* Mail the notebook to [markapurameswarkumar@gmail.com](mailto:markapurameswarkumar@gmail.com)

# **Details**

### Raw notes

* Capture detailed discussions and decisions
* Find the sub Treads in the given dataset

### Ideas for later

* Data preprocessing

**Offline Promotional Ideas**

1. Meme Paper Posting

Presenting the memes based on hackathon - introducing gdg hackathon and theme of the hackathon.

* 2 days before the hackathon
* Hackathon day

1. Balloon Spreading

Mention the hackathon's name and the hackathon's date on the balloons and spread those over the campus, especially on the ground and corridor.

* 1 day before the hackathon

1. Flash Mob by

By GDG Team.

- 3 days before the hackathon

1. Chocolate Promotion( 2nd year)

Chocolate distribution based on gdg-related questions for the 2nd year.

1. Domain Video Promotions

Explain the Domains to the students.

* 5 days before the hackathon

1. QR Codes Promotion

* 4 days before the hackathon

**HOD & Dean Permission - 10.02.2025 Monday**

**Titles:**

**GDG InnoHacks** - Finalized Title

GDG IntelliHack

GDG HackVerse

GDG ResoHack

GDG Hacks

GDG Learnathon

Web Dev

### **Web Dev**

**Incharges:**

1. **Sisindri Singamsetti - +91 9502414128**
2. **Dishika Vaishkiyar**

**🚀 Hey Code Warriors! 🚀**

**All the best for the hackathon! 🎯**

* **Code smart, debug fast! 🛠️**
* **Stay focused, think outside the box! 💡**
* **Work as a team, learn as you go! 🤝**
* **Keep your UI clean and your logic strong! 💻**
* **Don’t panic if you hit errors—every bug is a lesson! 🐞**

**May your code run smoothly, your APIs respond fast, and your ideas shine bright! ✨**

**🔥 Happy coding and best of luck! 🔥**

### **Question: The Invisible Job Board**

🔍 **web dev:**In an era where job seekers are inundated with an overwhelming number of listings—many of which do not align with their expertise—a groundbreaking hiring platform is introducing an innovative mechanism that **restricts access to job opportunities** unless candidates can substantiate their relevance. The philosophy behind this paradigm shift is that only individuals who demonstrably possess the requisite skill set should be granted visibility into potential career prospects.

Your objective is to architect a **highly interactive job portal** that strictly adheres to the following principles:

* **Complete Concealment:** Job listings must remain imperceptible by default, ensuring that users are unable to browse any positions without demonstrating competency.
* **Skill-Driven Discovery:** The portal should prompt users to manually input their proficiencies. As the user enters skills, job postings that correlate with at least one of the provided competencies should **gradually materialize** on the interface.
* **Relevance-Based Prioritization:** Any position exhibiting an alignment of **80% or greater** with the user's skill set should be prominently marked as a **"Top Match."**
* **Adaptive Ordering Mechanism:** Job postings must be arranged in a hierarchical fashion, with the most relevant opportunities surfacing at the forefront.

To enhance the sophistication and user experience of the system, consider integrating:

* **Intelligent Predictive Assistance:** A responsive input field that anticipates and suggests skills as users type, leveraging a precompiled repository of industry-relevant expertise.
* **Seamless Visual Transitions:** Implement elegant visual effects that orchestrate a **smooth unveiling** of job postings rather than an abrupt appearance.
* **Optimized Accessibility:** The platform should be meticulously engineered to ensure that its interface maintains fluid adaptability across a multitude of screen dimensions, particularly for mobile users.

❗ **Final Challenge:** At no point should users have the ability to view any job listings unless they actively specify at least one qualifying skill. The mechanism facilitating this revelation must operate autonomously, eliminating the need for explicit "search" interactions such as buttons or manual triggers.

🔑 **Considerations to Navigate the Challenge:**

* Evaluate how **dynamic filtering methodologies** can be leveraged to enhance data visibility while maintaining logical constraints.
* Contemplate an approach where **structured mapping of proficiencies to job descriptions** enables an intuitive yet computationally efficient matching process.
* Design an architecture where **real-time event-driven interactions** seamlessly govern the job list’s adaptability in response to user input.

This challenge necessitates a **nuanced understanding of frontend development principles**, compelling participants to think critically about **state management, input handling, and real-time UI updates**.

### **Roadmap for Building the Project**

#### Step 1: Setup the Project

* Choose a frontend framework (React.js recommended).
* Set up a development environment using Vite or Create React App.
* Install necessary dependencies (e.g., Tailwind CSS, Framer Motion).

#### Step 2: Design the UI

* Create a responsive layout using Tailwind CSS or Chakra UI.
* Design the input field for skill entry with autocomplete functionality.

#### Step 3: Integrate APIs

* Fetch job listings using the Indeed API or Adzuna API.
* Use the O\*NET Skills API or Hugging Face Transformers for skill matching.

#### Step 4: Implement Dynamic Filtering

* Use Lodash.js or RxJS to filter and sort job listings based on user input.
* Add a relevance-based prioritization system (e.g., 80% match = "Top Match").

#### Step 5: Add Animations

* Use Framer Motion or React Spring to animate the appearance of job listings.

#### Step 6: Optimize for Accessibility

* Ensure the platform is accessible using tools like Lighthouse and Axe.
* Test responsiveness on multiple devices.

#### Step 7: Deploy the Project ( Optional )

* Deploy the app on Vercel, Netlify, or Render.
* Test the live version for bugs and performance issues.

### **AI Tools for Specific Tasks**

#### 1. Skill Matching

* OpenAI GPT API  
  Use GPT to generate skill suggestions and match them with job descriptions.  
  <https://platform.openai.com/>

#### 2. Debugging & Code Optimization

* DeepCode  
  An AI-powered code review tool for identifying bugs and optimizing code.  
  <https://www.deepcode.ai/>

#### 3. Content Generation

* Copy.ai  
  Generate placeholder text or descriptions for job listings.  
  <https://www.copy.ai/>

### Alternative Approaches

* Voice-Based Skill Input  
  Use the Web Speech API to allow users to input skills via voice.  
  <https://developer.mozilla.org/en-US/docs/Web/API/Web_Speech_API>
* Local Storage for User Preferences  
  Save user skills and preferences in local storage for a personalized experience.  
  <https://developer.mozilla.org/en-US/docs/Web/API/Window/localStorage>
* Lottie Animations  
  Use Lottie for smooth and engaging animations.  
  <https://lottiefiles.com/>

### **🛠 Essential Resources & Tools for Development**

#### **1️. Frontend Frameworks & Libraries (For UI & State Management)**

Since this is a **frontend-only project**, you need tools to manage dynamic content, UI interactions, and responsiveness.

* **Primary Choice:**
  + [React.js](https://react.dev/) – Efficient for building dynamic UI with state management.
* **Alternatives:**
  + [Vue.js](https://vuejs.org/) – Progressive framework with built-in reactivity.
  + [Svelte](https://svelte.dev/) – Simple syntax and compiled at build time for better performance.

#### **2️. UI Styling & Component Libraries**

For building an interactive, modern, and responsive UI.

* **Primary Choice:**
  + [Tailwind CSS](https://tailwindcss.com/) – Utility-first CSS framework for fast UI development.
* **Alternatives:**
  + [Bootstrap](https://getbootstrap.com/) – Pre-styled components and grid system.
  + [Material UI](https://mui.com/) – Google's Material Design components for React.

### **🌐 APIs for Content Rendering**

To dynamically fetch **job listings, skills, and job descriptions**, you can use APIs to populate the page.

#### **3️. Job Listings APIs**

Fetching real-world job data dynamically.

* **Primary Choice:**
  + [Indeed API](https://www.indeed.com/developer) – Provides real-time job postings.
* **Alternatives:**
  + [LinkedIn Jobs API](https://developer.linkedin.com/docs/rest-api) – Retrieves job data from LinkedIn.
  + [Google Jobs API](https://cloud.google.com/talent-solution/job-search) – AI-powered job search API.

#### **4️. Skills Dataset & AI-Powered Search**

Since the challenge involves **predicting and matching skills dynamically**, these APIs will help:

* **Primary Choice:**
  + [O\*NET Skills API](https://services.onetcenter.org/) – Database of jobs, descriptions, and skills.
* **Alternatives:**
  + [Algolia API](https://www.algolia.com/) – AI-powered search API for skill suggestions.
  + [Skill List Dataset](https://github.com/tombatossals/skillicons) – Open-source tech skills database.

**5️. AI-Based Autocomplete & Predictive Assistance**

To suggest relevant skills in real time.

* **Primary Choice:**
  + [Claude AI](https://claude.ai/) – Can generate predictive suggestions for relevant skills.
* **Alternatives:**
  + [OpenAI GPT API](https://openai.com/) – Can be used for skill recommendation.
  + [Google Cloud Natural Language API](https://cloud.google.com/natural-language) – AI-powered text analysis.

### **🔄 Dynamic Content Handling & Rendering**

To process, filter, and display job postings dynamically.

#### **6️. JavaScript Libraries for Filtering & Sorting**

Since job listings should **dynamically appear based on input**, these will help:

* **Primary Choice:**
  + [Lodash.js](https://lodash.com/) – Provides filtering, sorting, and data manipulation utilities.
* **Alternatives:**
  + [Ramda.js](https://ramdajs.com/) – Functional programming for real-time filtering.
  + [D3.js](https://d3js.org/) – Can be used for visualizing job data.

#### **7️. Smooth UI Transitions & Animations**

Since job postings should gradually **appear smoothly**.

* **Primary Choice:**
  + [Framer Motion](https://www.framer.com/motion/) – Best for smooth animations in React.
* **Alternatives:**
  + [GSAP](https://greensock.com/gsap/) – Advanced animations for a polished UI.
  + [Animate.css](https://animate.style/) – Simple CSS-based animations.

### **🚀 AI Tools for Development & Debugging**

For faster coding, debugging, and content generation.

#### **8️. AI Code Assistants**

* **Primary Choice:**
  + [Claude AI](https://claude.ai/) – Helps generate optimized code, debug issues, and refine logic.
* **Alternatives:**
  + [ChatGPT](https://openai.com/chatgpt) – For generating code snippets and fixing bugs.
  + [GitHub Copilot](https://github.com/features/copilot) – AI-driven coding assistant.
  + [Codeium](https://www.codeium.com/) – Free alternative to Copilot.

#### **9️. Hosting & Deployment**

To make the project live.

* **Primary Choice:**
  + [Vercel](https://vercel.com/) – Best for **React-based** projects.
* **Alternatives:**
  + [Netlify](https://www.netlify.com/) – Best for static frontend hosting.
  + [GitHub Pages](https://pages.github.com/) – Free hosting for static sites.

### **Additional Resources & Tools**

#### 1. Frontend Frameworks & Libraries

* Solid.js  
  A declarative JavaScript library for building fast and reactive UIs.  
  <https://www.solidjs.com/>
* Preact  
  A lightweight alternative to React with similar API and performance benefits.  
  <https://preactjs.com/>

#### 2. UI Styling & Component Libraries

* Chakra UI  
  A modular and accessible component library for React.  
  <https://chakra-ui.com/>
* Styled Components  
  A CSS-in-JS library for styling React components.  
  <https://styled-components.com/>

#### 3. APIs for Job Listings

* Adzuna API  
  Provides job listings with salary trends and company insights.  
  <https://developer.adzuna.com/>
* USAJobs API  
  Official U.S. government job listings.  
  <https://developer.usajobs.gov/>

#### 4. Skills Dataset & AI-Powered Search

* Kaggle Skills Datasets  
  Open-source datasets for tech skills and job descriptions.  
  <https://www.kaggle.com/datasets>
* IBM Watson Natural Language Understanding  
  AI-powered text analysis for skill extraction.  
  <https://www.ibm.com/cloud/watson-natural-language-understanding>

#### 5. AI-Based Autocomplete & Predictive Assistance

* Hugging Face Transformers  
  Pre-trained models for natural language processing tasks like autocomplete.  
  <https://huggingface.co/transformers/>
* Rasa  
  Open-source AI for building contextual assistants.  
  <https://rasa.com/>

#### 6. Dynamic Content Handling & Rendering

* RxJS  
  A reactive programming library for handling asynchronous data streams.  
  <https://rxjs.dev/>
* Underscore.js  
  A utility library for filtering, sorting, and manipulating data.  
  <https://underscorejs.org/>

#### 7. Smooth UI Transitions & Animations

* React Spring  
  A spring-physics-based animation library for React.  
  <https://www.react-spring.io/>
* Popmotion  
  A functional animation library for JavaScript.  
  <https://popmotion.io/>

#### 8. AI Tools for Development & Debugging

* Replit AI  
  An AI-powered coding assistant integrated into the Replit IDE.  
  <https://replit.com/site/ai>
* Tabnine  
  An AI code completion tool that supports multiple languages.  
  <https://www.tabnine.com/>

#### 9. Hosting & Deployment

* Render  
  A cloud platform for deploying web apps with free tier options.  
  <https://render.com/>
* Surge.sh  
  A simple and free static site hosting service.  
  <https://surge.sh/>

### Useful Links

* React Official Documentation  
  <https://reactjs.org/docs/getting-started.html>
* Tailwind CSS Documentation  
  <https://tailwindcss.com/docs>
* Framer Motion Documentation  
  <https://www.framer.com/motion/>
* Indeed API Documentation  
  <https://developer.indeed.com/>

### **💡 Alternative Approaches**

If you want to make the project even more interactive: ✅ Use **Web Speech API** for **voice-based skill input**.  
 ✅ Implement **Local Storage** to remember **user preferences**.  
 ✅ Experiment with **Lottie animations** for **revealing job postings smoothly**.

These **resources, APIs, and AI tools** will help you build, **fetch job data, filter results, animate UI, and deploy the project** successfully. Let me know if you need help implementing any part of it! 🚀

# **Action items**

* **Complete the designing**
* **Complete within the deadline**

**Mail the notebook to** [**Sisindri Singamsetti**](mailto:sisindrisingamsetti@gmail.com)

UI/UX

## **UI/UX**

**Incharges:**

1. **K Gnana Sekhar**
2. **Shashank**

**🍽️ The Vanishing Flavor – A UI/UX Design Challenge for a Restaurant Ordering App**

A well-known **restaurant app** has experienced a strange issue—some dishes have mysteriously disappeared from the menu! Customers can see the restaurant and categories, but many items remain hidden.

The restaurant believes that only true food lovers can unlock the full menu. Your mission?

✨ Design an engaging and interactive restaurant app UI in **Figma**, where users can reveal hidden dishes, **switch between veg/non-veg**, and use a **"Swipe to Order"** feature.

**🛠 UI/UX Design Requirements**

Your app should have four key interactive sections:

**1️⃣ The Blurred Welcome & Menu Toggle**

The app opens with a blurred restaurant logo and tagline.

As users tap or scroll, the text gradually clears to reveal the restaurant’s identity.

A Veg/Non-Veg Toggle Button allows users to switch between categories smoothly.

**2️⃣ The Interactive Menu Section**

The menu appears in a list or grid view, showing a variety of dishes.

Some dishes appear locked or slightly faded until users interact (tap or swipe).

Clicking on a dish opens its detailed view screen.

**3️⃣ The Dish Details Screen**

When a user selects a dish, they see:

A high-quality image of the dish

Description & Ingredients

Customizable options (e.g., spice level, add-ons)

Price & estimated delivery time

Subtle parallax or zoom-in effects enhance the dish image.

**4️⃣ The Swipe to Order Feature**

At the bottom, instead of a regular button, a “Swipe to Order” bar appears.

When the user swipes right, a progress animation fills up, confirming the action.

A success screen appears, showing:

“Order Placed Successfully!”

Estimated delivery time

A small animation (e.g., confetti or a dish being served)

## **🛠 Essential Resources for UI/UX Design**

To successfully build this **visually immersive restaurant website** in **Figma**, you need access to the right tools and assets.

### **1️. UI/UX Design Tools & Prototyping**

* **Primary Choice:**
  + [Figma](https://www.figma.com/) – Best for designing and prototyping animations.
* **Alternatives:**
  + [Adobe XD](https://www.adobe.com/products/xd.html) – Good for **transitions & interactions**.
  + [Framer](https://www.framer.com/) – Great for **advanced UI motion effects**.

### **2️. Animation & Motion Effects in Figma**

Since the challenge demands **smooth, elegant animations**, these plugins will help:

* **Primary Choice:**
  + [Figmotion](https://www.figma.com/community/plugin/733025261168520714/Figmotion) – Adds **motion effects** directly in Figma.
* **Alternatives:**
  + [Anima](https://www.animaapp.com/) – Converts Figma designs into **live animations**.
  + [Smart Animate (Built-in)](https://help.figma.com/hc/en-us/articles/360040028613-Create-animations-with-Smart-Animate) – For smooth **page transitions** and hover effects.

### **3️. Interactive Hover & Scroll Effects**

To make elements **dynamically appear** instead of staying static.

* **Primary Choice:**
  + [Scroll Animations](https://www.figma.com/community/plugin/1040933320409380591) – Adds **interactive reveal effects**.
* **Alternatives:**
  + [LottieFiles for Figma](https://www.figma.com/community/plugin/739058006757759323/LottieFiles) – For **realistic motion graphics**.
  + [Jitter](https://jitter.video/) – Creates **custom micro-interactions** for UI elements.

### **4️. High-Quality Restaurant UI Kits & Icons**

For **faster** and **more elegant designs**.

* **Primary Choice:**
  + [Foodie UI Kit](https://www.figma.com/community/file/1035155559530723152) – Ready-made food UI components.
* **Alternatives:**
  + [Flaticon Food Icons](https://www.flaticon.com/) – Free **restaurant-themed icons**.
  + [Icons8](https://icons8.com/) – High-quality **SVG icons & illustrations**.

### **5️. Fonts & Color Palettes**

A **luxury restaurant website** needs **elegant fonts and warm, inviting colors**.

* **Primary Choice:**
  + [Google Fonts (Playfair Display)](https://fonts.google.com/specimen/Playfair+Display) – Perfect for **sophisticated branding**.
* **Alternatives:**
  + [Adobe Fonts](https://fonts.adobe.com/) – Exclusive **high-end typography**.
  + [Coolors](https://coolors.co/) – Helps generate **perfect restaurant color schemes**.

### **6️. AI-Powered UI Assistance**

For **enhanced creativity and automatic UI generation**.

* **Primary Choice:**
  + [Claude AI](https://claude.ai/) – Helps refine **visual storytelling & layout choices**.
* **Alternatives:**
  + [ChatGPT for UI](https://openai.com/chatgpt) – Can suggest **Figma structure & best practices**.
  + [Uizard](https://uizard.io/) – Converts **sketches into UI designs automatically**.

<https://www.toools.design/>

<https://creattie.com/lottie-animated-illustrations/all?type=all&orderBy=order&page=1>

<https://storyset.com/>

<https://www.figma.com/design/ciXabENxdIpwIQOLbhNxm7/FREE-Resources-by-GNANA?node-id=0-1&t=HgABcJEC8GZnCanB-1>

Note - Explore more inspiration sites through browsing no references were provided

### **⚡ Challenge Constraints**

✅ **All interactions should feel smooth and natural**—avoid jarring transitions.  
✅ **Use animations to guide the user, not distract them**.  
✅ **The design should be simple yet visually immersive**—focus on **storytelling through visuals**.  
✅ **Ensure responsiveness**—the design should adapt well to mobile screens.

# **Action items**

* **Complete the designing**
* **Complete within the deadline**

**Mail the notebook to** [**gnanasekharofficial@gmail.com**](mailto:gnanasekharofficial@gmail.com)

AI ( DS & ML )

**AI ( DS & ML )**

**Incharges:**

1. **A Anirudh**
2. **Greeshma**

**Attachments**

[train](https://docs.google.com/spreadsheets/u/0/d/1fU0ccNC6rBvAlcnBgm_rj3IzECgBW_rrDk3pdQEzVMQ/edit) Dataset in Attachment

# **Question**

* Imagine you're a newly hired Data Science Manager at a prestigious organization. You're tasked with developing an employee promotion prediction model. Your boss demands exceptional performance and emphasizes that subpar results will not be tolerated.

Here are the cryptic clues to guide your quest:

| **Topic** | **File** |
| --- | --- |
| * Isolate your development environment to prevent contamination. | venv |
| * Consult the sacred texts: Features briefing, Pandas, NumPy, Scikit-learn, TensorFlow, PyTorch, and Keras cheat sheets. | [pytorch-cheatsheet-en.pdf](https://drive.google.com/open?id=1h5ZeNSkyuwkMRuPGE-TiN6YlR_dLTpvI) [Pandas\_Cheat\_Sheet.pdf](https://drive.google.com/open?id=15c8gPo-8gijc8omuXrLF621BXsOMMGWi) [TensorFlow2Cheatsheet.pdf](https://drive.google.com/open?id=1XQAHMB8YJ6U6sav6KUHhY_Tln0o7-owo) [Keras\_Cheat\_Sheet\_gssmi8.pdf](https://drive.google.com/open?id=1pyZweRBCX3swH4qcqZpyeV0oC_SJPT5r) [Numpy\_Python\_Cheat\_Sheet.pdf](https://drive.google.com/open?id=1B0ZU4l69InR_CafK8sRnFkF0EtU0ZDVR) |
| * Seek divine inspiration from Google and ChatGPT. |  |

# **Summary**

* At last the model should predict the output
* Any algorithm can be used
* Mode of approach should be mentioned in very first of the notebook 📒

# **Action items**

* ****Complete the modelling
* Complete within the deadline
* Mail the notebook to [alevooruanirudh@gmail.com](mailto:alevooruanirudh@gmail.com)

# **Details**

### Raw notes

* Capture detailed flow of the modeling
* Predict the output
* Markdown the entire process step by step
* The accuracy score and confusion matrix score will be evaluated

Power Bi

**Power Bi**

**Incharges:**

1. **M Eswar Kumar**
2. **A Anirudh**

**Attachments**

* [citation.txt](https://drive.google.com/open?id=1r8tCX7VwkF9Rvf9AiP7unUKLXcD2oRAs)
* [IHME-GBD\_2021\_DATA-eef9b0d7-1.csv](https://drive.google.com/open?id=12Y-KpWWfzHm5_x9D4AnwtNIPkILVgwfo)

# 

# **Question**

* Analyze the Impact of Viruses on Mortality Rates Globally.

#### **Problem Statement:**

* Participants are tasked with creating a Power BI report to visualize and analyze the causes of death due to viruses. They should identify key insights about the mortality rates, trends, and regional impacts of specific viruses. The focus should be on presenting actionable insights using compelling visualizations.

| **Challenge Question:** | **Summary** |
| --- | --- |
| Top Causes of Death | * Identify and visualize the leading viruses causing death. |
| Trend Analysis: | * Show the trend of deaths caused by viruses over the years. Is there an increase or decrease? |
| Correlation Insights | * Explore potential correlations between mortality rates and other factors (if available in the dataset). |

# **Summary or key decisions**

* Summarize the most important outcomes
* Capture the trends/patterns

# **Action items**

* Complete the dashboard
* Complete within the deadline
* Mail the notebook to [markapurameswarkumar@gmail.com](mailto:markapurameswarkumar@gmail.com)

# **Details**

### Raw notes

* Capture detailed discussions and decisions
* Find the sub Treads in the given dataset

### Ideas for later

* Data preprocessing