Research: Building a High-Quality MTG AI Deck-Builder Website

Overview

This project aims to create a Magic: The Gathering (MTG) deck-builder website powered by AI. The goal is to make it the **best deck builder available** by combining a clean user experience, effective AI assistance and monetization strategies. To achieve this, we analysed user feedback and reviews of existing deck builders (TappedOut, Archidekt, Moxfield, Deckstats, Scryfall, ManaStack, MTGGoldfish, etc.), investigated API limitations (Scryfall, Cardmarket, TCGplayer and OpenAI), studied interface design principles and color schemes, and explored multi-agent AI frameworks. The findings below are intended to inform both the developer and the AI that will assist users when using the system.

User Feedback on Existing Deck Builders

Usability and interface

- **Outdated or cluttered interfaces hurt adoption** – players criticised TappedOut for its dated look, poor UI and lack of auto-complete or deck analysis 【269423912428179†L157-L183】. MTGGoldfish also received complaints about an unpleasant UI and the need to reload pages to edit details 【269423912428179†L217-L241】. Users praised Moxfield as the most modern-looking and intuitive deck builder 【269423912428179†L344-L380】 and described Archidekt as cleaner than Moxfield but with inferior playtester and deck-builder overlays 【269423912428179†L186-L215】.

- **Drag-and-drop and card grouping are essential** – comments in CompetitiveEDH threads indicated that drag-and-drop category assignment should be a staple feature; many users build decks in Archidekt because it supports this interaction 【809167502580443†L320-L324】 【809167502580443†L335-L339】. Moxfield's inability to drag cards into categories and hidden search functions were common complaints 【809167502580443†L813-L851】.

- **Mobile compatibility matters** – some players build decks on their phone. Users said Moxfield is smoother on mobile while Archidekt struggles 【986387452154683†L1088-L1096】. Others use ManaBox or Deckstats for mobile deck tracking 【986387452154683†L1088-L1096】.

- **Auto-complete & fuzzy search** – Scryfall's fuzzy search (recognising misspelled card names) is loved 【269423912428179†L244-L276】, and users noted that Archidekt and Moxfield both support Scryfall syntax search; however, search features are sometimes hidden or require memorising syntax 【986387452154683†L1113-L1143】 【809167502580443†L813-L851】.

- **Deck analysis & statistics** – Deckstats stands out for strong analysis tools: curves, card-type distributions and probability calculations 【269423912428179†L320-L339】.

ManaStack and Moxfield offer some analysis but not as in-depth 【269423912428179†L272-L305】. Users appreciate features like showing which tokens are needed 【986387452154683†L1254-L1256】.

- **Playtesting features** – players like deck testers that allow drawing hands, casting spells and simulating turns. Moxfield's playtester is intuitive but limited to one player 【269423912428179†L344-L380】, while Archidekt's playtester is cluttered 【269423912428179†L186-L215】. Deckstats includes a starting-hand simulator 【269423912428179†L320-L339】.

- **Collection management and price integration** – Archidekt slows down with large collections and is capped at ~200 unique cards because the UI loads too many images 【612101342725984†L42-L96】. Users want easy import/export of card lists and integration with card scanners or collection managers (DragonShield scanner, ManaBox) 【192999953128504†L686-L766】.

- **Community & social features** – players enjoy Moxfield's social features (primers, ability to share lists and comment) 【986387452154683†L1166-L1170】. TappedOut and other older sites have outdated comment systems and limited community tools.

- **Legal deck checks & AI reliability** – in a Reddit thread about an AI deck builder (KrakenTheMeta), users requested features to mark illegal cards and highlight duplicates or cards outside the intended format. The developer responded by adding a status indicator and button to view illegal cards 【192999953128504†L344-L394】. Users also wanted the AI to import lists (via CSV or from their collections) and ensure the generator uses legal cards only 【192999953128504†L510-L591】 【192999953128504†L686-L766】.

Summary: critical improvements

1. **Modern, responsive interface** with an intuitive drag-and-drop deck editor, clearly visible search bar and fuzzy auto-complete. Ensure mobile-friendly design and support for dark/light modes.
2. **Robust deck analysis tools** (mana curve, colors, synergies, probability calculators, starting-hand simulators). Highlight needed tokens and recommended land counts. Provide warnings when cards are illegal in selected formats.
3. **Flexible collection management** – support import/export via CSV or integration with scanners (ManaBox/DragonShield). Allow generating decks from the user's collection. Avoid performance bottlenecks by loading card images lazily and limiting DOM complexity 【612101342725984†L42-L96】.
4. **Comprehensive playtesting** – include multiplayer playtesters or at least two-sided simulation. Provide hand-draw simulation and in-game actions.
5. **Community features** – support comments, primers, deck sharing and rating. Consider personal profiles, following other users and tagging or starring favorite decks.
6. **AI-assisted suggestions** – integrate AI to propose card upgrades, synergies, deck names and play lines. Provide transparency about how recommendations are generated and allow editing suggestions.

Differentiating Your AI Deck Builder

The project already aims to combine a ChatGPT-powered assistant with Scryfall card data, but there are opportunities to make the tool **unique** compared with existing platforms and AI models.

Unique feature ideas

- **Multi-agent AI orchestration** – rather than relying on a single large model, use a *team* of specialised agents (e.g., a "researcher" agent to fetch card data, a "strategist" agent to design the deck, a "rules-lawyer" agent to check legality, and a "reviewer" agent to explain the deck). Frameworks like **CrewAI** provide role-based collaboration where agents share context and build upon each other's contributions 【897843549012520†L107-L124】. **LlamaIndex agents** combine retrieval-augmented generation with knowledge bases to answer data-heavy queries 【897843549012520†L171-L190】. **AutoGen** enables asynchronous conversations among multiple agents, useful for long tasks where the AI must wait for external events 【897843549012520†L125-L147】. Leveraging these frameworks can produce more reliable and transparent deck recommendations.

- **Contextual deck building** – allow the AI to consider a user's **collection** (by uploading a list or scanning via APIs) and build decks using owned cards first. Add toggles such as "budget deck", "competitive", "for beginners" or "themed around X". Use large language models to interpret natural language prompts ("build me a budget Elves deck using only cards from Modern Horizons and my collection").

- **Integrated price and availability checks** – incorporate real-time price data from card markets (subject to API permissions) to suggest decks within a budget. Because Cardmarket and TCGplayer limit API access (Cardmarket prohibits new apps and restricts data usage 【761737800472170†L9-L25】 【171539230944312†L18-L23】, TCGplayer no longer grants new API access 【171539230944312†L18-L23】), rely on Scryfall price estimates (which are allowed for personal use), affiliate links or third-party price services such as JustTCG. Provide warnings that prices are estimates.

- **Explainable AI** – accompany AI-generated decks with explanations: why each card is chosen, how the mana curve works, synergy notes and potential upgrades. This helps users trust the AI and learn from it. Provide button to ask "why did you include this card?" or "what would you change to reduce cost?".

- **In-browser simulation** – generate sample draws and allow the AI to narrate typical turns ("turn-by-turn guidance"). Use small animations or interactive card UI (discussed below) to make this engaging.

- **Adaptive search** – design a search bar that accepts plain language ("two-cost green creatures that draw cards") and Scryfall syntax. Provide search help or auto-suggestions, avoid hiding search behind menu tabs 【809167502580443†L813-L851】.

- **Flexible deck views** – let users switch between text list, card-image view, color-categorised columns or price view. Provide toggles for dark/light mode and allow customizing accent colors or background patterns (within pre-approved themes).

How your product can stand out from ChatGPT and other AI tools

- **Domain-specific knowledge base** – incorporate MTG rules, format legality and card rulings into the assistant via retrieval-augmented generation. Use Scryfall's bulk data (legalities, oracle text) and wotc bulletins. ChatGPT alone may hallucinate card details; adding curated knowledge reduces errors.

- **Focus on deck optimisation rather than just generation** – generic chat models may propose random lists; by combining data on metagame win rates, EDHREC statistics and synergy scoring, your assistant can provide targeted suggestions (e.g., substituting underperforming cards). Provide analytics that ChatGPT cannot natively produce.

- **Personalisation and community** – allow AI to remember user preferences (preferred colours, decks owned, typical budgets) and adjust recommendations accordingly. Build a community where users can share AI-generated decks, provide feedback and contribute prompts. ChatGPT does not offer persistent, shareable deck profiles.

Monetization Strategies and Growth Plan

Monetizing an MTG AI deck builder requires balancing user accessibility with sustainable revenue. Consider the following staged approach:

Stage 1 – Early launch / user growth

1. **Free access with optional donations** – start with a free tier to attract users. Accept donations via Patreon or Ko-fi; Moxfield's Patreon offers tiers (£1–£39/month) that remove ads and provide Discord access 【326085603722853†L34-L93】. A similar model could fund server costs while keeping the core product free.
2. **Affiliate marketing** – join card retailer affiliate programs (Cardmarket, TCGplayer, JustTCG) and include affiliate links when users view or purchase cards. Since Cardmarket's API has strict terms (no new applications and content must manage your own inventory 【761737800472170†L9-L25】 【108179939072172†L347-L362】) and TCGplayer has stopped granting new API access 【171539230944312†L18-L23】, rely on general affiliate links rather than API calls. Ensure compliance with their rules.
3. **Display ads** – integrate targeted Google AdSense or other ad networks on non-premium pages to generate passive income. Follow best practices to avoid clutter.

Stage 2 – Paid features / subscriptions

1. **Premium membership** – offer a subscription that unlocks advanced AI capabilities (e.g., unlimited deck recommendations, advanced analytics, saved search preferences), removes ads and provides early access to new features. Tiers can start at a low price (£2–£5 per month) and scale up with additional perks (Discord roles, vanity URLs) similar to Moxfield 【326085603722853†L34-L93】.

2. **Data-intensive features** – heavy API usage (card price updates, multi-agent calls) can be reserved for paying users to offset costs. Provide a free quota per day and charge for additional calls (microtransactions) while abiding by API rate limits.
3. **Merchandise and services** – once the brand is established, sell digital goods (custom card sleeves or playmats design files), coaching sessions (AI-assisted deck clinics), or premium content (video guides, webinars). This extends beyond the core deck builder and leverages the community.

Stage 3 – Enterprise or partner offerings

1. **White-label solutions** – license the AI deck-builder API to local game stores or tournament organisers so they can embed it on their sites. Provide custom theming and analytics; charge a monthly fee.
2. **Advertising / sponsorships** – partner with content creators, podcast hosts or event organisers to feature the deck builder. Sponsor tournaments or challenge events; cross-promote with game accessory companies (e.g., sleeves, dice makers).

Key monetization notes

- Always offer a **usable free tier** to retain community goodwill. The MTG community is sensitive to aggressive monetization; treat premium features as value-add rather than gating basic deck building.
- Respect API terms. Cardmarket and TCGplayer restrict data usage and new developers 【761737800472170†L9-L25】 【171539230944312†L18-L23】. Use Scryfall's free bulk data (keep calls under 10 requests per second and include proper headers 【962996004932226†L50-L93】), and rely on publicly available price information or partner programs rather than scraping.
- Consider Google's evolving ad products. Google's Offerwall allows readers to watch an ad or answer a survey to access premium content – this could let users unlock AI-generated deck analyses without paying.

Visual Design and Customisation

A successful deck builder needs to look modern, be easy to scan and feel fun. Key design principles and research insights are summarised below.

Card-based UI for deck lists

The **card UI pattern** is widely used in modern interfaces because it groups related information into modular containers that are easy to scan and rearrange 【827028447476523†L145-L190】. Advantages include:

- **Improved navigation & scannability** – cards break down content into bite-sized pieces, letting users quickly find what matters 【827028447476523†L173-L195】.
- **Responsiveness** – cards adapt to different screen sizes, re-flowing from multi-column layouts on desktops to single columns on mobile 【827028447476523†L185-L188】.
- **Versatility** – they can contain images, buttons and text, making them ideal for deck lists, card suggestions and analytics 【827028447476523†L189-L196】.

The article's design principles recommend:

1. **Focus each card on one thing**; avoid cramming too much information 【827028447476523†L291-L297】.
2. **Organise information clearly** with a hierarchy (title, main message, supporting details) 【827028447476523†L298-L303】.
3. **Use a grid system and consistent spacing/alignment** for a clean look 【827028447476523†L309-L321】.
4. **Make cards feel interactive** through hover effects, shadows and making the whole card clickable 【827028447476523†L325-L337】.
5. **Choose simple, easy-to-read fonts** and ensure the design is responsive across devices 【827028447476523†L347-L358】.

Applying these principles, deck lists can be displayed as draggable cards that show the card image, name, cost, type and options to increase/decrease count or view details. Cards could have quick actions (e.g., "remove" or "replace") accessible via icons.

Color schemes and branding

A cohesive color palette improves brand recognition and user experience. The Hostinger tutorial advises designers to understand their brand and audience, learn basic colour theory and test combinations; using tools like Adobe Color or Coolors can help select complementary palettes 【870313862082279†L164-L178】. Colour schemes should enhance readability and evoke desired emotions 【870313862082279†L139-L163】. The popular **60–30–10 rule** (60 % dominant colour, 30 % secondary, 10 % accent) can guide choices. For example, MTG's guild colours (blue and black for Dimir, green and white for Selesnya) can inspire themes.

User customisation and autonomy

The Nielsen Norman Group notes that allowing users to make **surface-level customisations** (e.g., theme colours, light/dark mode, switching list vs grid views) can give a sense of ownership and delight 【168632214277916†L100-L120】. However, most users do not heavily customise interfaces, and designers should **maintain consistency and brand recognition** 【168632214277916†L100-L114】. Offer a few pre-validated themes with accessible contrast, and allow advanced users to adjust accent colours or view modes. Provide toggles for card image sizes, text font size and column arrangement, but keep the core workflow consistent. Additional customisation options (e.g., keyboard shortcuts, saved layouts) can be added for power users 【168632214277916†L120-L142】.

API Access and Potential Pitfalls

Your site will rely on external data sources (Scryfall, card markets, AI models), each with its own limitations. It is critical to respect these restrictions to avoid being blocked or violating terms of service.

Scryfall API

- The API is free for personal use but has a **rate limit of ~10 requests per second**; repeated or misconfigured requests can trigger 400 or 403 errors 【962996004932226†L50-L93】. To avoid this:
  - Use Scryfall's **bulk data endpoint** to download entire card lists and cache them locally rather than making per-card requests.
  - Add proper **User-Agent and Accept headers** and use HTTPS for all requests 【962996004932226†L50-L93】.
  - Do not funnel all requests through a single IP, and avoid "exhaustive searches across all values" which can lead to IP bans 【962996004932226†L50-L93】.

Cardmarket API

- **No new API applications** are being accepted 【761737800472170†L9-L25】. Existing applications may not share credentials; misuse or scraping can result in revocation 【761737800472170†L9-L25】. Data is meant for managing the user's own inventory; publishing price data without permission violates the terms 【108179939072172†L347-L362】.
- Dedicated apps have daily request limits (5 000 for private users, 100 000 for commercial users, 1 000 000 for "power sellers") 【807154659803161†L104-L112】. If you plan to use Cardmarket data, you must become a registered seller and apply for an app key. Consider partnering with a seller or using the public website for price display with attributions.

TCGplayer API

- **TCGplayer has stopped granting new API access** and existing endpoints are no longer maintained 【171539230944312†L18-L27】. New developers cannot obtain keys; you must rely on affiliate links or third-party alternatives (e.g., JustTCG) for price information.

OpenAI / other AI APIs

- Large language models like GPT-4 have **rate limits measured in requests per minute (RPM) and tokens per minute (TPM)**. For example, the default GPT-4 tier allows 3 500 requests/minute and 90 000 tokens/minute; exceeding this triggers 429 errors 【349164313813591†L74-L104】. Implement caching, queueing and token optimisation; reduce long prompts and summarise intermediate results. The AI assistant should detect when the API is nearing its limit and gracefully degrade (e.g., temporarily restrict AI suggestions or queue tasks).
- If using multiple AI providers (Anthropic, Mistral, etc.), check each provider's rate limits and pricing, and design the multi-agent system to route tasks to the appropriate model based on cost and performance.

Caching & offline functionality

- To minimise API calls, cache card data, price snapshots and AI responses. Use server-side caching and a local database; refresh daily or as allowed by the provider. Provide an offline mode for previously fetched collections so users can build decks even during outages.

Multi-Agent AI Workflows

Implementing a multi-agent AI system can make your deck builder more robust and modular. The Langfuse blog summarises several open-source frameworks and when to use them 【897843549012520†L125-L147】 【897843549012520†L147-L190】:

- **LangGraph** – represents tasks as nodes in a directed acyclic graph (DAG) and is suited for complex workflows requiring explicit branching and error handling 【897843549012520†L44-L63】. Use this if your deck building pipeline needs to handle multiple decision points (e.g., budget vs competitive paths).
- **OpenAI Agents SDK** – an official framework for building agents that can call functions/APIs and reason about tasks. It integrates naturally with OpenAI models 【897843549012520†L64-L80】.
- **Smolagents** – a minimal, code-centric agent loop for small automation tasks; useful for quick calculations or generating Python code to manipulate card data 【897843549012520†L84-L100】.
- **CrewAI** – emphasises role-based collaboration among multiple agents; a "crew" coordinates specialists (researcher, strategist, reviewer) to solve a problem 【897843549012520†L107-L125】. This fits the deck builder scenario where different agents handle searching cards, constructing lists, checking legality and explaining decisions.
- **AutoGen** – supports asynchronous, multi-turn conversations among agents 【897843549012520†L125-L147】. Suitable for long interactions, e.g., when the AI must wait for a user to upload a collection or confirm choices.
- **Semantic Kernel & LlamaIndex** – provide enterprise-level orchestrations and retrieval capabilities 【897843549012520†L150-L168】 【897843549012520†L171-L190】. Use these if the site needs to integrate with external systems (e.g., business processes, knowledge bases).

When designing your AI assistant, choose a framework based on complexity and required collaboration. A **CrewAI-style** solution with a planner agent delegating to card search and deck construction sub-agents can produce more consistent decks while allowing each agent to specialise. Monitor agent interactions via tools like Langfuse to debug and improve workflows.

Recommendations for Implementation

1. **Design a clean, responsive UI** that uses card-based components, clear typography and consistent spacing. Provide drag-and-drop editing, visible search with fuzzy auto-complete and toggleable dark/light themes. Keep the deck editor intuitive on mobile and desktop.
2. **Implement robust deck analytics**: mana curves, colour balance, card type distribution, probability simulations and token lists. Allow the AI to highlight problem areas and suggest replacements.

3. **Use multi-agent AI frameworks**: orchestrate specialised agents (e.g., card search, strategy builder, rules validator, price analyst) and implement memory and caching to comply with API limits. Provide explainable outputs.
4. **Offer customisation without compromising usability** – give users a few high-contrast themes, font-size adjustments and view modes. Save preferences in user profiles. Avoid letting users break the layout or stray from brand guidelines.
5. **Introduce community features early**: deck comments, ratings, follower systems and the ability to publish primers and guides. Encourage user-generated content and integrate social sharing.
6. **Plan monetization in phases**: start with a free service supported by donations and affiliate links; later introduce memberships with advanced AI features, ad-free experiences and Discord perks; eventually license the technology to third parties. Maintain transparency and avoid paywalling essential functions.
7. **Respect all API terms**: throttle Scryfall requests, avoid scraping Cardmarket, and abide by TCGplayer's restrictions. Where API access is unavailable, rely on publicly available data or affiliate links instead of scraping. Use caching to reduce network calls.
8. **Test and iterate**: gather user feedback during beta; use analytics to understand feature adoption; adjust UI, AI suggestions and monetization based on real usage.

Conclusion

Building an AI-assisted MTG deck-builder website presents both opportunities and challenges. Users demand modern, mobile-friendly interfaces with drag-and-drop functionality, powerful search and analysis, and strong community features. Many existing platforms fall short in these areas. By combining robust UI design, retrieval-augmented AI models and multi-agent workflows, your project can deliver personalised, explainable deck suggestions that stand out from generic chat-based solutions. Careful attention to monetization (without alienating users) and strict compliance with API limitations will ensure sustainability. Continuous iteration based on user feedback will be key to making this the **best deck builder available**.