Endstate Features before launch.

1. Built in local model inference server:
   1. Ability to pull down recent models from hugging face
   2. Have an interface that allows users to pull in models that fit in memory
2. Build in tab predictions for sentence completions: (Done)
   1. Use a small (ish)model that can nicely and intuitively predict the next sentence you'll type.
   2. As you move on it gets rid of new contact
3. Build in tab predictions for next action prediction:
   1. Where youll go next
   2. As your typing all it to predict changes eg grammatical
4. Build in CMD K:
   1. Have it pull in super local context of what around where the cursor currently is
   2. Have it pull in specifically what is highlighted
   3. Show diffs with a nice CMD Y / CMD N or CMD + enter to accept
   4. Allow for follow up questions before acceptance

Nice features i want. When you make an edit from chat it then scrolls to the diff in the page.

Autocomplete could be improved by having an embedding being made on the prediction and if its too similar to teh current or previous sentence not have it

1. Deeply understand the domain and CUJs,

2. Segment out the system into "problems that traditional software is good at solving" and "LLM-hard problems",

3. For the LLM hard problems, work out the smallest possible state space of decision making,

4. Build the system, and get users using it,

5. Gradually expand the state space as feedback flows in from users.

Think harder button

What makes a code editor a code editor?  
  
1. Intelligent Code Completion

2. Integrated Debugging Tools

3. Version control

4. Smart Navigation - Features like "Go to Definition," "Find All References,"

provide the types of things that novel writers have access to. E.g., the ability to keep an outline summarizing the chapters, notes on each character, and the ability to try multiple experiments to see which set of plot twists results in the most interesting resolution.