I spent some time thinking about what applications we should start to try to implement. Many applications are still woefully underspecified, so I think we should choose based on the dual criteria of relevance and clarity of existing specification documents. I've come up with the following list, on which I now want to ask the Juvix team for feedback.

- Multichat
- Described in an upcoming blog post by Patrick (ask him).
- The main things to implement here would be channels, permissions, and messages.
- Described in an upcoming blog post by Patrick (ask him).
- The main things to implement here would be channels, permissions, and messages.
- Promise Graph
- Specified here.
- The main concept would be to implement tasks as described in this document as resources, with update rules as specified, and token payments conditional on task completion.
- Specified here.
- The main concept would be to implement tasks as described in this document as resources, with update rules as specified, and token payments conditional on task completion.
- Knowledge Graph (w/CRDTs)
- Described a bit here.
- The main things to implement here are CRDTs, permissions, and doubly-linked pages.
- Described a bit here.
- The main things to implement here are CRDTs, permissions, and doubly-linked pages.
- Kudos
- Described a bit here.
- We can start simple - with kudo denominations, liquidity positions, swaps, automated flow / transfers - and build more functionality over time.
- Described a bit here.
- We can start simple - with kudo denominations, liquidity positions, swaps, automated flow / transfers - and build more functionality over time.
- Public Signal
- Described a bit here.
- This depends on parts of Kudos.
- Described a bit here.
- This depends on parts of Kudos.
- Chess Tournament
- Described a bit here.
- The basic concept is that users can play chess matches against each other, where Anoma enforces the rules of chess, permissions of who can play what moves, timed clocks, etc. Users can also stake tokens before a game, and the winner takes the whole pot. Future application versions could integrate tournaments, ratings, and more complex multi-game rules.
- Described a bit here.
- The basic concept is that users can play chess matches against each other, where Anoma enforces the rules of chess, permissions of who can play what moves, timed clocks, etc. Users can also stake tokens before a game, and the winner takes the whole pot. Future application versions could integrate tournaments, ratings, and more complex multi-game rules.

- Type-Checked Git
- Described a bit here.
- We could start with a very very simple language (not Juvix itself - like the simply-typed lambda calculus, say) as a demonstration of the concept.
- Described a bit here.
- We could start with a very very simple language (not Juvix itself - like the simply-typed lambda calculus, say) as a demonstration of the concept.

Juvix folks - @paulcadman @jan @Lukasz - let me know what you think, especially what's clear here and what isn't in terms of knowing approximately what to implement.