Xeni - Flights Refactor Proposal

Scope of work Phase-1

1. Xeni-API

Current Problems

1. Currently the backend architecture initiates a new websocket connection with CTW on every flight search.
2. There is a lot of repeated CTW related code with translates to business logic scattered through out the app
3. Promises hell (This is a programming term used to describe deeply nested promises)
4. Eslint screams with errors
5. Unmaintainable and bad coding practices with about 2500 lines of backend code. No Classes, No Library etc.
6. Xeni API and Xeni UI uses vanilla Websocket connections instead of using a library/framework.

Proposed Solution

1. Refactoring the code into more manageable concerns
2. Extracting out the CTW communication bridge into it's own class/library
3. Making sure that the communication is established between CTW and Xeni Server just once so that we don't exceed the 150 connections hard limit.
4. Fix all the ES lint issues
5. Extract out business logic to Models/Library where ever possible.
6. Switch to use socket.io to communication between the xeni-frontend and API

Note:

1. No new APIs would be added
2. No new features would be added
3. The before and after would look the same excepting that the underlying code would be better managed.

2. Xeni-UI

Current Problems

1. Search tool which is the heart of the app is very badly written.
2. All the services (flights, hotels, transfers) is one single component
3. Containers routes are deeply nested in component tree.
4. Eslint screams red due to errors.
5. mixed used of inline, local and global CSS
6. Vanilla Websocket library is used instead of depending on a framework.
7. Few components are 5k lines big, break it down into smaller manageable pieces.

Proposed Solution

1. Refactor about 25 files (already shared by Ayush).
2. Strictly follow Container, Pages, Component patterns
3. Move inline and global CSS into local CSS
4. Fix eslint issues
5. Migrate from vanilla websocket to Socket.io
6. Rewrite the flight logic with bad coding practices, eg:
    a. Single character variable
    b. For loops
    c. Multiple render through out the component.

Note:

1. Other than socket.io no other libraries would be touched/upgraded in this phase.
2. Unit/Integration tests is not included in this phase.

Timeline of this refactor

This refactor would be completed in 4 weeks, i.e, 15th of May.

Final Goals

1. At the end of this refactor, we should have a codebase that is neat and manageable by future developers.
2. This refactor needs to be carried out in master so that Shopify Marketplace can benefit and pull the refactored code from master.