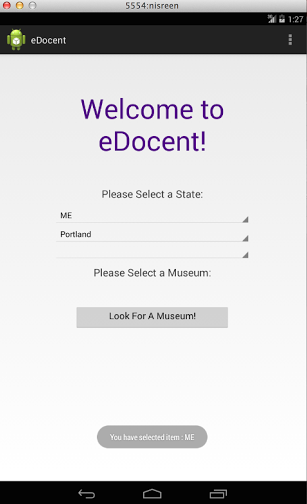
**eDocent Android Application Documentation**

**User Interface**

*Splash Page*

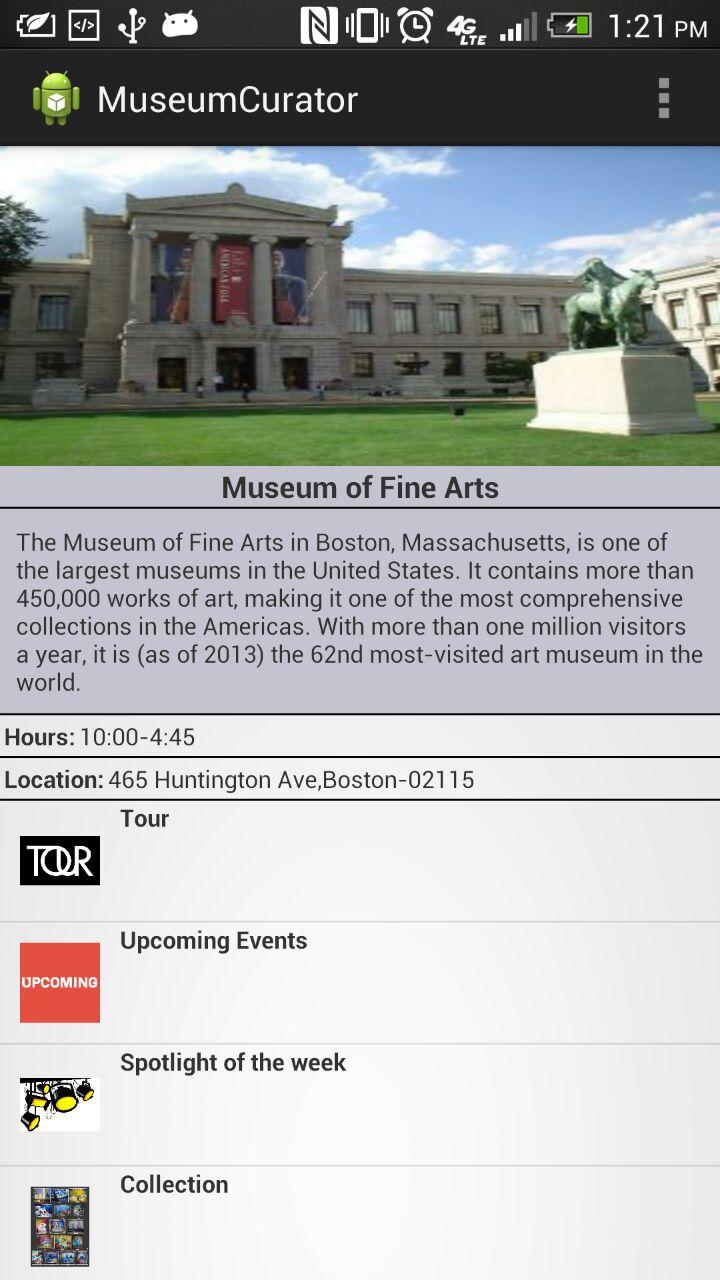
In the splash page, the user is shown three spinner objects, drop down menus. The first drop down menu will prompt them to select a state, the second a city/town, and finally the third will prompt them to select a museum that will be in the state and city/town they selected. All these menus will be dynamically generated by API calls. Once the user selects a museum, they will be directed to that museum’s page in the Android application.



**Fig1. Splash Page**

*Museum Home Page*

The museum home page will display a picture of the museum, a small blurb about the museum, information such as museum hours, address, current exhibitions, upcoming events, spotlight of the week, collections, etc. If they click on collections, they will be able to view the art that the museum currently has. Users can learn more about specific art pieces at the museum by clicking on them under the collections list. This information will be given by the museum.



**Fig2. Museum Home Page**

*Art Page*

Art pages are what display information about a specific piece of art in the museum. Users will be able to read more about the piece of art than what is displayed at the museum. Each art piece will also include the style of art, period in which it was made, and the artist. This information will be given by the museum.

In the Android application, the user will only be giving input in the homepage. Otherwise, they are only able to see whatever the museum uploads and shows them. They cannot edit any museum or art information within the mobile application.

**Behind the Scenes**

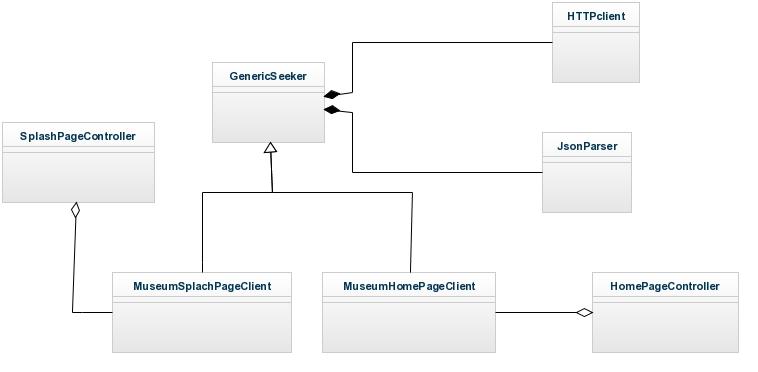
All the information the mobile application needs is stored in the database. The mobile application calls an API, which queries the database and gives the requested data to the mobile application. The data is in JSON, therefore, the mobile application parses the data and then displays the information to the user.

When an application is launched, we call a GUI thread for the user view. We do not call the server to retrieve data using the same thread, if we would use the same thread, the application will hang. Hence we create a new background thread to fetch data from the server.

**Architecture**

We use the Model-View-Controller architecture for implementing the user interface.

In the below UML diagram we give a brief description of the controller activities. The GenericSeeker class is the base class which contains the HTTPclient and the JsonParser. Each page would have its own controller. For example the Museum Splash Page Client class would fetch data from the database. The Splash Page Controller has a pointer to the data model. The data model is populated and we can view the information in the user interface.



**Fig3. Controller UML Diagram**