GitHub Username: jorjSB

Articles Registry

Description

Document, share, save and read offline articles as old as 1851. Powered by The New York Times.

Intended User

Any historian, student, teacher or just curious George can obtain news related to any subject as old as 1851 for academic purposes or just general knowledge, to satisfy one's curiosity or just a good conversation topic.

Features

- Accesses online, verified information;
- Saves data for offline (based on the user preferences and his latest accessed data);
- Offers a widget;
- Offers share capabilities;
- Supports tablet(large displays) with adapted UI;
- Supports RTL layout switching;
- Includes support for accessibility;
- Material design UI;
- Displays ads;

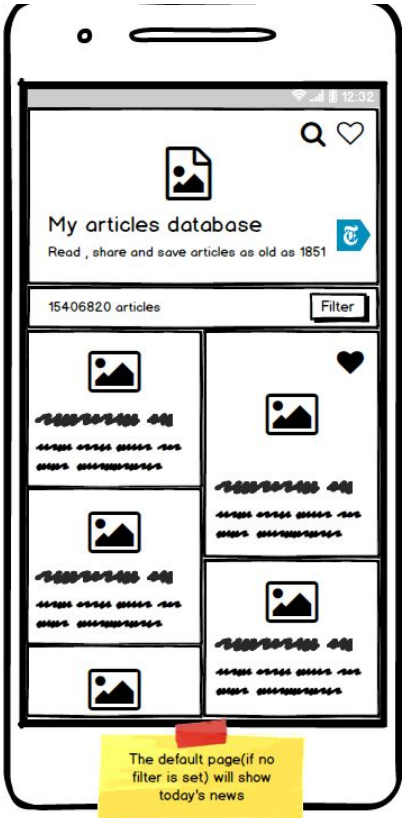
User Interface Mocks

These can be created by hand (take a photo of your drawings and insert them in this flow), or using a program like Google Drawings, www.ninjamock.com, Paper by 53, Photoshop or Balsamiq.

1 - Main app view

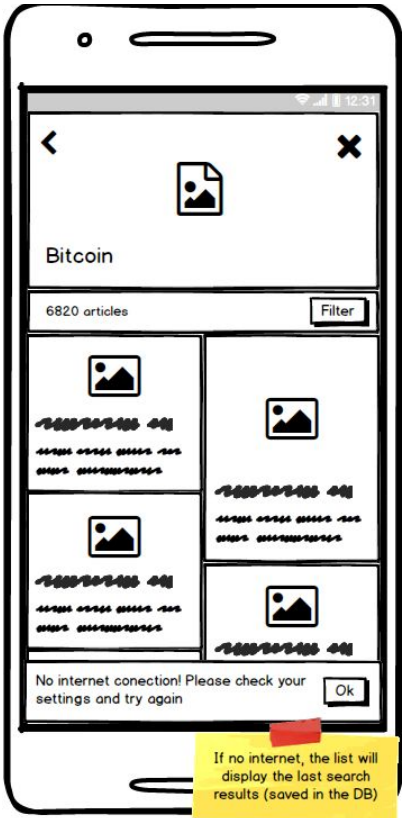
Splashscreen - short app description and credentials. Dissapears after 5 seconds.

2 - Main app view



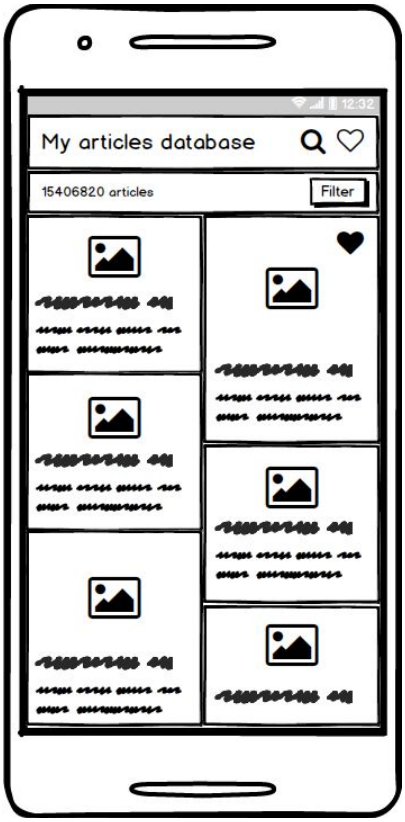
Main app view: extended App Bar (following material design principles), list with today's articles(initially), if no search or filters were applied, snackbar if no internet.

3 - Main app view



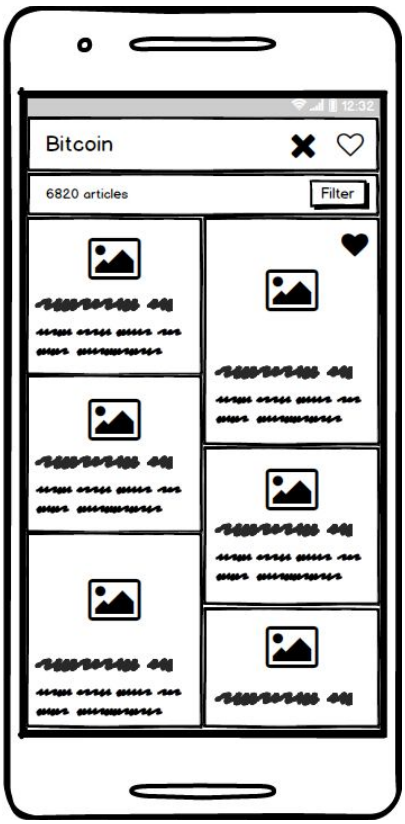
Main app view: displays results after a search was made. This case: “bitcoin”.

4 - Main app view



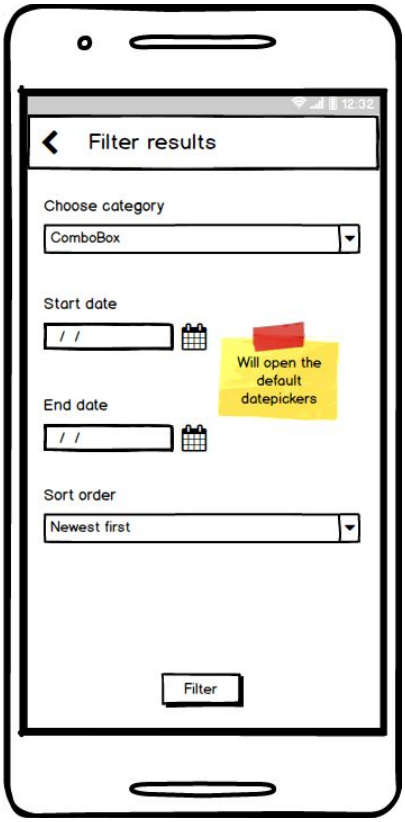
Main app view: app bar is collapsed;

5 - Main app view



Main app view: displays results after a search was made. This case: “bitcoin”. User started scrolling, app bar is collapsed;

6 - Filtering/order

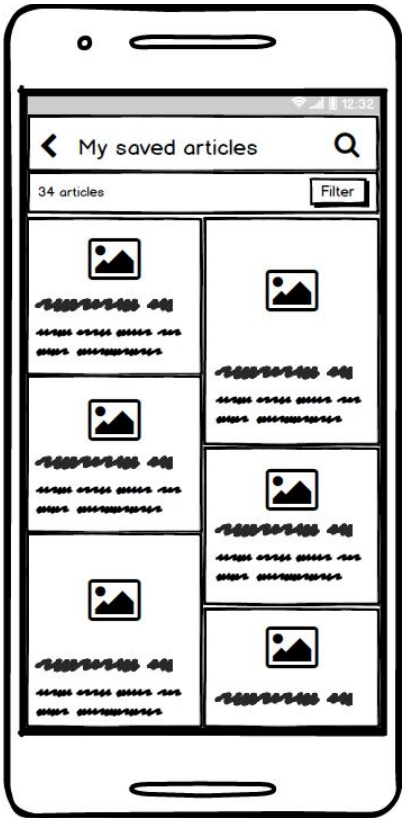


Filter view. The search can be refined by categories, start/end date and ordered. When the filters will be applied, and a new query will be made.

7 - Main app view

Main app view: user’s saved articles.

8 - Main app view



Main app view: user’s saved articles, AppBar collapsed.

9 - Main app view

Main app view: user’s saved articles, no article saved.

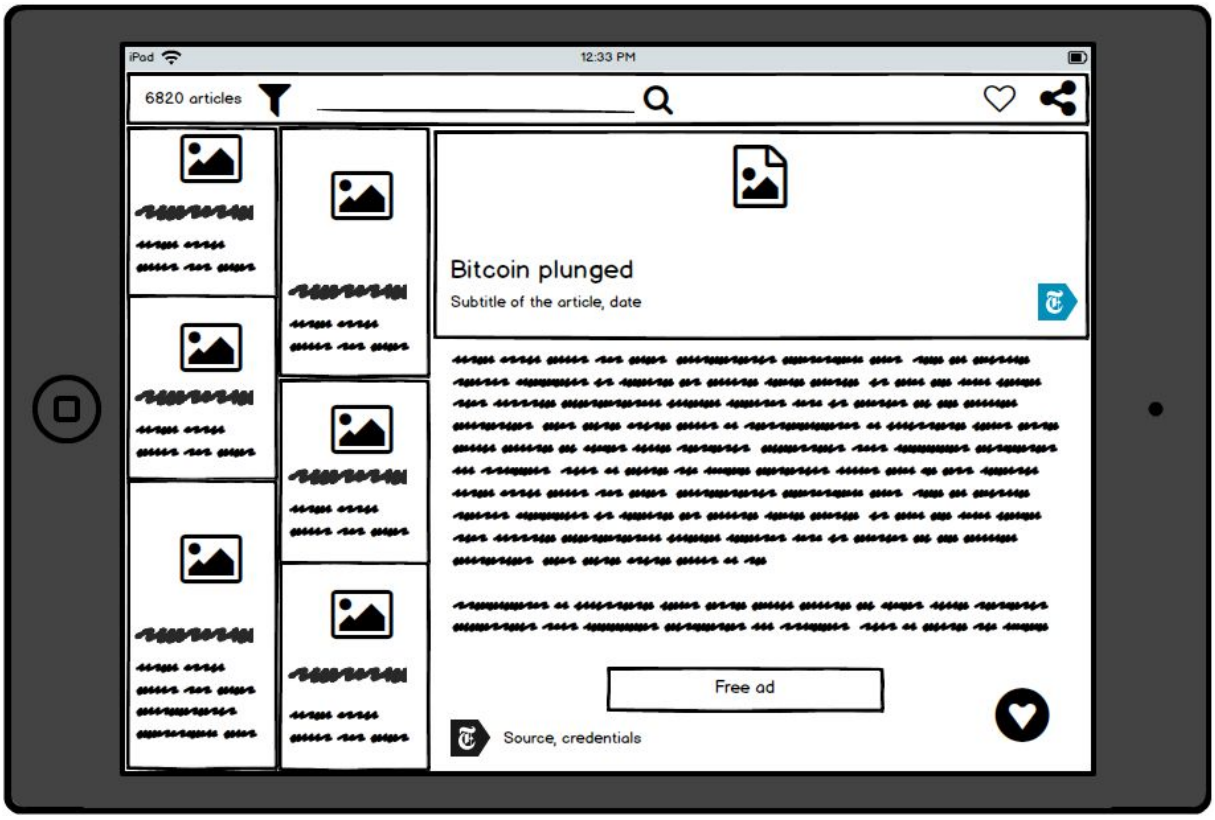
10 - Article description

Article description: collapsible toolbar, FAB for offline saving, snackbar for feedback, share button and credentials.

11 - Article description

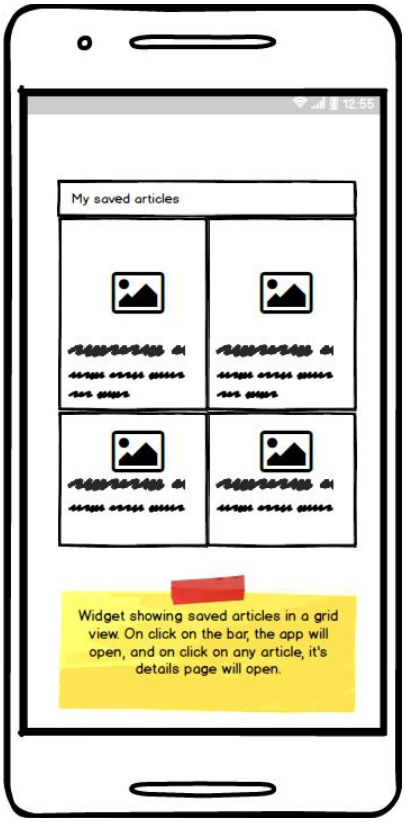
Article description: AppBar collapsed.

12 - Large screen display



Large screen display: master/detail UI design pattern applied. (*Balsamiq only offers iPad layout for large screens)

13 - Widget



Widget: displays the user's saved articles. Will open the app(click on bar) or the individual article(click on article item).

Key Considerations

How will your app handle data persistence?

Room will be used(along with its LiveData and ViewModel) for storing/accessing/deleting the articles.
Whenever a new set of data(articles) is fetched, it will be parsed and stored in the DB(except in the case of pagination), and the entire DB will be erased, except the articles that were flagged as "saved" by the user.
On access, the last accessed data(from the DB) will be shown to the user.

Describe any edge or corner cases in the UX.

The navigation is pretty straight-forward: from the main page(list with articles), the user can navigate to the article's detail page or the filtering page, and return using back button.

This will not apply on the large screen layout: the back navigation will not be necessary as the "main list" and the other pages are always displayed or changed from the navbar or the articles list.

Describe any libraries you'll be using and share your reasoning for including them.

- Gson - to parse Json objects for storing into the DB;
- Room (persistance) - DB orm;
- Picasso - handle images caching etc.
- butterknife - for binding views, cleaner code;
- okhttp - handles network requests;
- android Design library - support for developing a Material Design UI.

Describe how you will implement Google Play Services or other external services.

1. I will implement Google Analytics so I can have a clear picture of the user's behaviour: saves, shares, number of searches, searched terms..
2. Admob - will display test ads. In the case of such an application, some revenue might be obtained, so Admob is a good starting place.

Next Steps: Required Tasks

This is the section where you can take the main features of your app (declared above) and break them down into tangible technical tasks that you can complete one at a time until you have a finished app.

Task 1: Project Setup

Create a new project in Android Studio and GitHub;

- Set the required initial permissions (like internet access);
- Add in Gradle the known libraries I will use (listed above);

- Add the main activity (for my project I think it is enough one activity, since I will use Fragments in order to obtain the Master/detail UI design pattern for large screens);

Task 2: Fetch and parse/store the data

I will start working on the main subject of the app: retrieving, storing and displaying articles;

- Start working on the data model for an article(a PJO);
- Use httpOk library to configure a class that handles the network queries and fetches data using the API (http://developer.nytimes.com/);
- Use Gson lib. to parse the data, handle missing data cases etc.
- Setup Room so I can store the data;

Task 3: Display the data

After fetching and storing data, Il'l create a simple RecyclerView to display it.

- Setup the master fragment that will be used to display fetched articles;
- Use Room's LiveData and ViewModel classes to listen/retrieve data from the DB and display it in a RecyclerView
- Handle the pagination (only 10 results are served one time, so pagination logic will have to implemented: while scrolling, before the end of the list is reached a new call will be made, and the results will be stored into the DB and added in the RecyclerView);
- Handle orientation changes;
- Use Picasso to display images;

Task 4: Search/filter/order

I will need to add a basic search dialog in the header and another fragment for setting the filtering/ordering if the user desires so.

- Create a search Dialog and link it with the class that handles the network queries;
- In case of a new successful search, all the articles from the DB(except the ones saved by the user) will be deleted, and the new fetched ones will be stored/displayed;
- Create a new custom fragment or dialog that handles the filer/order. I will create a class that provides the filter values(categories that may interest the user) that will be displayed in a spinner, another spinner for the sorting by date and 2 additional fields for the start date and the end date. Those will use the android's default Calendar View to set the dates. When the user submits the form, another call will be made with the selected values as described in the API's documentation. The new URL will be created and

handled like before: in the NetUtils class, the response will be parsed and stored in the DB and displayed.

Task 5: Details page

Setup the fragment that handles the details page.

- Create a new fragment;
- Handle the fragment transactions in the MainActivity.
- Setup the FAB, and the basic layout;

Task 6: Save articles for offline use and display them

I want to offer the users the option to save articles for offline use.

- Link the fab with a method that marks the article in the DB as “saved” or favourite;
- Handle the “unsave/ delete from DB case”, triggered by the same button;
- Handle the data replacement in the DB (when a new set of data is fetched, it will replace the entire DB, except the “saved” articles);
- Handle the access of those (saved)articles. The saved articles list can be accessed by a button in the AppBar, on the Master Screen.
- Handle the display/messages of the saved articles list;

Task 7: Add the “Splashscreen”

- I want to show the user a screen(another Fragment) before he enters the app in which I explain briefly where the data is fetched and what the app does;
- Add the options on the splashscreen to dismiss it or to hide it forever(a boolean in SharedPreferences);

Task 8: Start working on the UI

- Start working on the AppBars, handle transitions, elevations, style the app;
- Add the NY Time’s logo on the articles, as stated in the Terms and Conditions.
- Handle all orientation changes;
- Create and implement the fragment transition logic for the master/detail flow;
- Create other App Bar that will be used for large screens;

- Add SnackBars for giving user feedback on his actions: searches, filters, save an article, unsave an article, share, no internet connection etc.

Task 9: Add the free Ads

- Add the library if not yet added;
- Create/link the test account. project and what else is needed;
- Modify the layouts to display the ads in the detail page(s);

Task 10: Add analytics

- Add the library if not yet added;
- Create/link test account, project and what else is needed;
- Add the events wherever needed;
- Test that every event is sent in the console as I set it up to;

Task 11: Add widget

The widget will display the a list with the user's saved articles.

- Create the layouts, classes etc.;
- Handle the click events(open the app/article details);
- Handle the update from the DB: when a new article is added/deleted, the widget has to be updated. No automatically (scheduled) update is needed;

Task 12: Final UI polishing, tests etc. + Prepare for submission

- Check again the transitions;
- Manually test all the elements and functionality of the app and widget;
- Check the Large Screen dissplay;
- Orientation changes etc.
- Accessibility UI enhancements;
- Configure `installRelease task;`
- Add a signing configuration, add keystore/password;
- Push to git.

Submission Instructions

- After you've completed all the sections, download this document as a PDF [File → Download as PDF]
    - Make sure the PDF is named "**Capstone_Stage1.pdf**"
- Submit the PDF as a zip or in a GitHub project repo using the project submission portal

If using GitHub:

- Create a new GitHub repo for the capstone. Name it "**Capstone Project**"
- Add this document to your repo. Make sure it's named "**Capstone_Stage1.pdf**"